T.C.
SELÇUK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

MALZEME İHTİYAÇ PLANLAMA SÜRECİNDE PARTİ-HACİMLENDİRME PROBLEMİNİN KARAR DESTEK SİSTEMLERİ İLE ÇÖZÜMLENMESİ

Alihan GÜZELDÜLGER

YÜKSEK LİSANS TEZİ

ENDÜSTRİ MÜHENDİSLİĞİ ANABİLİM DALI

KONYA, 2009

İÇİNDEKİLER

Sayfa No:

Sayfa No:

ÖZET

YÜKSEK LİSANS TEZİ

MALZEME İHTİYAÇ PLANLAMA SÜRECİNDE PARTİ -HACİMLENDİRME PROBLEMİNİN KARAR DESTEK SİSTEMLERİ İLE ÇÖZÜMLENMESİ

Alihan GÜZELDÜLGER

Selçuk Üniversitesi Fen Bilimleri Enstitüsü

Endüstri Mühendisliği Anabilim Dalı

Danışman: Doç. Dr. Turan PAKSOY

2009, 91 Sayfa

Jüri: Doç. Dr. Turan PAKSOY
Prof. Dr. Ahmet PEKER
Yrd. Doç. Dr. A. Alpaslan ALTUN

Malzeme ihtiyaç planlaması, ana üretim planı, malzeme fişi, ürün ağacı, stok durumu bilgileri gibi girdilere ve sipariş önceliklerinin planlama raporları, performans kontrol raporları, temin süreleri gibi çıktılara sahip olan bir sistemdir. Bu tez çalışması, değirmen makineleri üretimi yapan İMAŞ A.Ş.'de malzeme ihtiyaç planlama sürecinde parti-hacimlendirme problemini bir karar destek sistemi yaklaşımı kullanarak çözümlemek amacıyla yapılmıştır. Bunun için literatürdeki sipariş miktarını belirleyen ve maliyet analizi yapan yöntemler araştırılmış ve bu yöntemler işletmede seçilen makine tipi için uygulanmıştır. Karar destek sistemleri, kararın yapısal olmadığı durumlarda karar alma işlemine yardımcı olmak için tasarlanmış, esnek bilişim teknolojisi sistemleridir ve bu tez çalışmasında karar destek sistemi olarak Visual Basic programından yararlanılmıştır. İşletmeye ait aylık talep miktarları girilip program işletildiğinde maliyeti minimize eden "periyodik sipariş miktarı" metodu en uygun yöntem olarak bulunmuştur. Burada ilk dönemden başlanarak kendisi dahil bir sonraki dönemin siparişi birlikte verilirken, bir dönem sipariş verildiğinde bunun hemen ardındaki dönem sipariş verilmemektedir.

Anahtar Kelimeler: Malzeme İhtiyaç Planlama, Parti-hacimlendirme, Karar Destek Sistemleri, Vals Makinesi.

ABSTRACT

MS Thesis

Resolving of the Lot-sizing Problems
In the Material Requirement Planning Process
By Decision Support Systems

Alihan GÜZELDÜLGER

Selçuk University

Graduate School of Natural and Applied Sciences

Department of Industrial Engineering

Advisor: Assoc. Prof. Dr. Turan PAKSOY

2009, 91 Pages

Jury: Assoc. Prof. Dr. Turan PAKSOY

Prof. Dr. Ahmet PEKER

Assist. Prof. Dr. A. Alpaslan ALTUN

Material requirement planning is a system that has inputs like master production scheduling, material receipt, product tree, inventory status information and outputs like planning reports of order precedence, performance control reports and lead time. This thesis study was made for the purpose of resolving the lot-sizing problem in material requirement planning process by using a decision support system approach in İMAŞ Incorporated Company that manufactures millers. Therefore, methods that determine order quantities and make cost analysis were researched and these methods were implemented for the machine selected in the company. Decision support systems are flexible information technology system that was designed to help the decision taking system in case the decision isn't structural and in this thesis Visual Basic programme was utilised. When monthly demand quantities inserted then the programme was run, "the periodic order quantity" that minimises the cost was found as the feasible method. Here beginning with the first period, the order of the present period and the order of the next period are given together and when an order was given in a period the next period no order would be given.

Key words: Material Requirement Planning, Lot-sizing, Decision Support Systems, Roller Mill.

1. GİRİŞ

Bu çalışmanın amacı, malzeme ihtiyaç planlama (MRP) sürecinde parti-hacimlendirme probleminin ele alınıp literatürdeki sipariş belirleme yöntemleri yardımıyla bu problemin değirmen makineleri üretimi yapan İMAŞ A.Ş.'de bir karar destek sistemi yaklaşımı ile çözümlenmesi dolayısıyla maliyet açısından en uygun yöntemin belirlenerek belirsizlik altında birtakım kısıtlarla üretim planı için gerekli matematiksel programlama modelinin geliştirilmesidir.

Yapılan tez, malzeme ihtiyaç planlaması sisteminin girdilerini-çıktılarını, faydalarını-sakıncalarını, karar destek sistemlerinin özelliklerini, literatürde yer alan on adet sipariş miktarı belirleme yöntemlerini ve değirmen makineleri üreten bir işletmede MRP sürecinde parti-hacimlendirme probleminin bir karar destek sistemi ile çözümleme uygulamasını, birtakım kısıtlar altında geliştirilen programlama modelini ve en uygun sipariş miktarı modeli için kazanç ölçütlerini kapsamaktadır.

Çalışma sonunda, uygulamanın yapıldığı işletmeye en az maliyetli sipariş miktarı belirleme yöntemi bulunmuş olacak, ne sıklıkla ve ne miktarda sipariş verilmesi gerektiği tespit edilmiş olacak ayrıca belirsizlik durumunda nasıl bir model geliştirilebileceği ortaya konmuş olacaktır.

Malzeme ihtiyaç planlamasının amacı, bütün envanter birimleri bazında brüt ve net ihtiyaçların belirlenerek etkin bir stok yönetimi için bilgi üretmesidir. MRP ile planlanmış ve kontrol edilmiş envanterler – planlanan imalatı ve sevkıyatı yapabilmek için malzemeler tesise istenildiğinde ulaştırılmış olur. Malzemelerin zamanında işletmede olması ile en az envanter sistemde bulundurulmuş olur. Ayrıca bu sistem ile hem üretim hem de satın alma açısından temin planları geliştirilir ve malzemelerin elde edilebilirliği ve teslimat zamanları hakkındaki en taze verilere dayanarak, çizelgeleme ve denetim fonksiyonları için öncelikler tespit edilmiş olur. Planı yapılan siparişlerden yola çıkılarak kapasitenin de planlanması yapılabilir (Acar 1999).

Malzeme ihtiyaç planlaması şu sebeplerden ötürü etkili bir stok kontrol sistemidir;

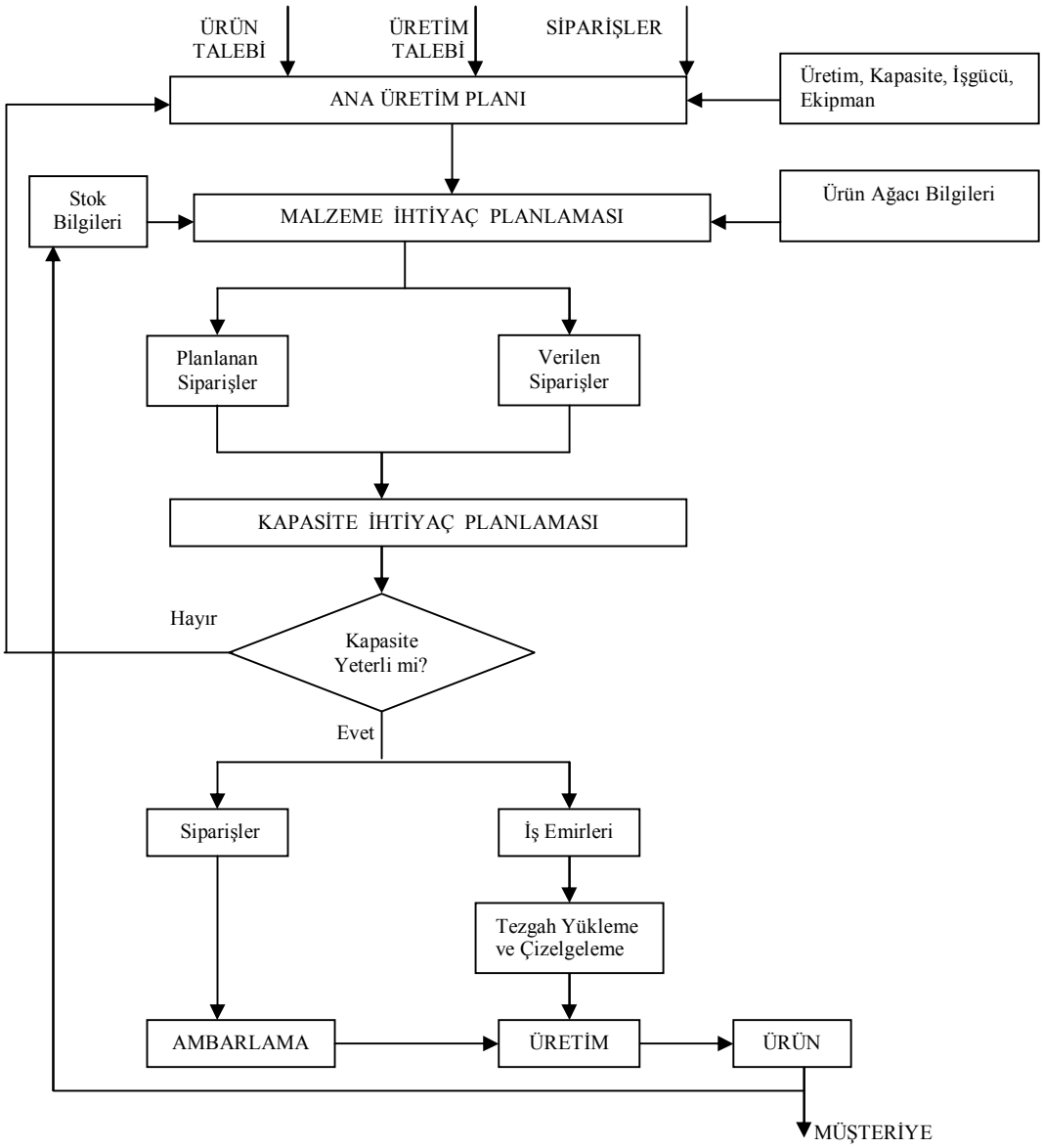
- Envanter yatırımları en alt seviyede tutulur,
- MRP sistemi değişimlere karşı esnektir,
- Sistem, stok birimleri bazında geleceğe yönelik bir bakış açısı ortaya koyar,
- Sipariş miktarları taleplere göre belirlenir,
- MRP sistemi, taleplerin zamanlaması ve tamamen karşılanması hususlarına özen gösterir (Acar 1999).

MRP'nin amaçları, planlanan ürünün imalat ve sevkıyatının zamanında sağlanması, hangi parçanın ne zaman satın alınacağı, parçanın bulunabilirliği, teslim tarihleri ile ilgili güncel bilgilere dayanarak çizelgeleme ve kontrol yapmak, kapasite planını yönetmektir. Özetlemek gerekirse malzeme ihtiyaç planlaması güçlü bir stok/imalat kontrolü, satın alma/sevkıyat planlama sistemidir (Çelikçapa ve Sarsılmaz 1999).

MRP'de sistemin amaçları şu şekilde sıralanabilir;

- Planlanan ve kontrolü yapılan envanterlerin, planlanan üretimi ve sevkıyatı hayata geçirebilmek için malzemelerin tesise zamanında gelmesini sağlamak,
- Malzemelerin istenilen anda firmada hazır bulunmasını sağlayarak sistemde en az miktarda envanter tutmak,
- Üretim, sevkıyat ve satın alma faaliyetlerinin planlanması, üretim ve satın alma açısından temin planlarının oluşturulması ve güncel hale getirilmesi, çizelgeleme ve kontrol fonksiyonlarının güncel verilere dayandırılması,
- Planlanan siparişleri yönlendirerek kapasite planının ortaya çıkarılması,
- Envanter birimleri bazında brüt ve net ihtiyaçların belirlenmesi ve bu yol ile gerçekçi bir stok yönetimi için veri oluşturulmasıdır (Çetinkaya 1988).

Şekil 1.1'de malzeme ihtiyaç planlama sisteminin yapısı gösterilmiştir.



Şekil 1.1. Malzeme İhtiyaç Planlaması Sistemi (Şahin 2003)

2. KAYNAK ARAŞTIRMASI

Ho ve ark. (2007), tarafından yapılan çalışmaya göre Malzeme İhtiyaç Planlaması (MRP), bağımlı talebe sahip sipariş stoklarını çizelgelemeyi yönetmek için bilgisayarla işlenmiş bir bilgi sistemidir. MRP'nin konusu genelde doğru parçayı, doğru miktarda ve doğru zamanda almaktır. MRP sistemleri 20 yy.ın sonlarında üretim tesislerindeki hammadde ve bileşenlerin akışını yönetmede çok önemli bir yaklaşım haline gelmiştir. MRP'nin esas odak noktası, bağımlı talep parçaları için etkin bir envanter yönetimi sağlamaktır. MRP sistemlerinin amacı, doğru zamanda, doğru sipariş miktarını belirlemek için doğru envanter bilgisini üretmektir.

Tek seviyeli kapasitelendirilmemiş parti-hacimlendirme problemi için Wagner Whitin (1958), değişken talep durumlarını en uygun biçimde çözmek için bir dinamik programlama prosedürü ortaya koymuştur. Yine de Wagner Whitin algoritması, matematiksel yapısının kompleks olması nedeni ile pratikte önemli derecede uygulanmamıştır. Haddock ve Hubicki (1989), 263 imalat firması üzerinde MRP yazılımında 11 parti-hacimlendirme modelinin kullanım sıklığı konulu anketi yapmışlar ve sezgisel parti-hacimlendirme yöntemlerinin, Wagner Whitin algoritmasının hiç kullanılmadığı durumda uygulandığı kanısına varmışlardır. Sezgisel tekniklerin bir başka önemli ve sık bahsedilme nedeni, Wagner Whitin algoritmasından planlama ufkunda genelde daha üstün olmalarıdır. Araştırmanın büyük çoğunluğu parti-hacimlendirme problemini çözmek için sezgisel teknikleri geliştirmeye yönlendirilmiştir.

Silver ve Meal (1973), ortalama hazırlık ve stok maliyetini minimize etmek için genelde Minimum periyot maliyeti (MPM) olarak bilinen bir sezgisel teknik önermiştir. Hu ve Munson (2002), 10 yayınlanmış nümerik çalışmayı incelemiş ve MPM'nin en fazla tavsiye edilen model olduğu kararına varmışlardır. Dahası, MPM'nin en çok umut verici olarak görünen sezgisel tekniklerden deneysel faktör ayarlarının geniş çeşitliliği altında test edilmiş iki metottan biri olduğunu belirtmişlerdir. Pan (1994), talep örneği, stok ve hazırlık maliyeti faktörlerini kullanarak altı yaygın sezgisel tekniğin ayrıntılı bir duyarlılık analizini uygulamıştır.

MPM'nin parametre itibariyle belirsizliğe en duyarsız sezgisel metot olduğu kadar hassas çözümler üretmede en iyi sezgisel teknik olduğu sonucuna varılmıştır. Saydam ve Evans (1990), Wagner Whitin algoritmasının en iyi bilinen dört sezgisel tekniklerle göreceli performans karşılaştırması üzerinde geniş ölçüde bir hesaplama çalışması yapmışlardır. Hesaplanan sonuçları MPM'nin hem çözüm kalitesi hem de hızı açısından en iyi performansa sahip olduğunu göstermiştir. MPM'nin ortalama performansı Wagner Whitin algoritmasının verdiği en uygun maliyetten %1,6 sapma göstermiştir.

DeMatteis (1968), sezgisel Parça periyodu algoritmasının (PPA) geliştirilmesi ile uğraşmıştır. Parça periyodu algoritmasında, zamandaki talep miktarı, partideki stok maliyeti sabit hazırlık maliyetine eşit veya ondan küçük olduğu sürece, parti-hacimlendirme miktarı $Q(i)$'ye eklenir. Baker (1989), PPA (-) olarak ifade ettiği, talepleri partideki stok maliyeti hazırlık maliyetinden kesinlikle az olduğu sürece biriktiren alternatif bir durdurma şartı içeren parça periyodu algoritmasının alternatif bir versiyonunu önermiştir. Diğer popüler parti-hacimlendirme prosedürleri: Gorham'ın (1968) Minimum toplam maliyet (MTM), Berry'nin (1972) Periyodik sipariş miktarı (PSM), Minimum birim maliyet (MBM), Ekonomik sipariş miktarı (EOQ) ve parti için parti (L4L) yöntemleridir. Baker tarafından önerilen ve MTM (-) olarak ifade edilen sezgisel minimum toplam maliyetin bir değişkeni, ardışık iki veya daha fazla periyotlar, stok ve hazırlık maliyetleri arasında eşit farklılıklara sahipken periyotlardan daha az sayıda olanı seçer (MTM periyotlardan daha fazla sayıda olanı seçerken).

Ho ve ark. (2007), tek-seviyeli kapasitelendirilmemiş durum için, bilinen iki MPM-tabanlı sezgisel parti-hacimlendirme metotları Minimum Periyot Maliyeti veya nMPM ve MPM'in gelişmiş bir versiyonu nMPM(i)'yi önermişlerdir. MPM algoritmasında uygulanan ortalama periyot maliyeti (OPM) kavramı planlama ufkundaki periyotların sayısı ile toplam maliyeti bölmeyi kapsarken, nMPM toplam maliyetin sıfır-olmayan talep periyotları sayısına oranı olan Net ortalama periyot maliyeti (NOPM) tabanlı sezgisel tekniktir. NOPM'nin kullanımı sipariş kapsamını uzatmaya veya planlama ufkundaki siparişlerin toplam sayısını azaltmaya dolayısıyla sıfır taleplerin meydana gelme durumunda maliyet performansını artırmaya öncülük eder.

Ho ve ark. (2007), sezgisel metotlarını mevcut yedi sezgisel metotlarla karşılaştırmak için MPM'ni içeren bir simülasyon çalışması uygulamışlardır ve geniş bir alandaki deneysel şartlarda her ikisinin de üstün ve güçlü performans sağladıkları sonucuna varmışlardır.

2.1. Malzeme İhtiyaç Planlama Sistemi Ve Tarihsel Gelişimi

Şenel'in (1996), yaptığı çalışmaya göre günümüz koşullarında, malzeme eksikliği, yükselen envanter taşıma maliyetleri vb. gelişmeler, daha yoğun kontrol ve yeniliğe daha çabuk uyum sağlamayı gerektirmektedir. Bu hedeflere ulaşmada yararlanılan malzeme ihtiyaç planlaması, yatırımları en aza indirmek, imalatı ve etkinliği artırmak ve müşteri memnuniyetini üst seviyelere çıkarmak için kullanılan bir çizelgeleme ve denetim metodudur. MRP, envanter yönetiminde bağımlı talep şartının olduğu durumda geçerlidir. Bu nedenle, MRP sisteminin ana kuralı olarak malzeme, parça ve ham maddelere olan talep, nihai ürüne olan talebe bağlıdır. MRP, planlama işlemine nihai ürünlerin tamamlanması gereken tarihten başlayarak geriye doğru gider, talebi bağımlı olan parçaların sipariş miktarlarını ve sipariş zamanlarını bulur. Bağımlı ürünlerin talebi, malzeme ihtiyaç planlama sistemi ile ana üretim planı sonuçlarına göre elde edilir. Tabi bunun sağlanması için talebi bağımlı olan malzemelerin yada parçaların istenen zamanda kullanıma hazır halde bulunması gerekir. İşte malzeme ihtiyaç planlaması bu işlevi yerine getirir.

İlgenli'nin (1989), yaptığı çalışmaya göre MRP, ana üretim planından yola çıkarak, üretimin planlanması ve aynı anda kontrol edilmesidir. Böylelikle stok seviyelerini azaltmak, müşteri ilişkilerini daha iyi yönetmek, talebi zamanında karşılayabilmek, ana üretim planını sürekli güncel tutmak, hazırlıkların ve atıl kapasitelerin neden olduğu maliyetleri azaltmak, yöneticilerin programın pratikteki katkılarını daha net görmesi mümkün olmaktadır.

Çelikçapa'nın (2000), yaptığı çalışmaya göre MRP seri üretim yapan işletmelerde;

- Stok maliyetlerinin düşürülmesi,
- Envanter kontrolünde insan kaynağı kullanımı azaltılır,
- Kapasite ayarlamaları kolaylaştırılır,
- Bu sistem ile değişimlere karşı daha kolay uyum sağlanır.

Kobu'nun (1999), çalışmasına göre MRP yönetiminde ana ilke, bağımsız talebe sahip bitmiş üründen geriye doğru giderek gerekli malzeme ve parçaları lazım olduğu anda kullanıma hazır etmektir. Bu prensip, stok kalemlerinin depoda bekleme süresini ve elde bulundurma maliyetlerini azaltır. Örneğin, gelecek ay montajı planlanan 1000 adet çamaşır makinesinin hemen siparişini vermek yerine montajdan birkaç gün önce siparişini vermek çok daha avantajlı olacaktır.

Şenel'in (1996), yaptığı çalışmaya göre MRP sistemi, ana üretim planını zaman boyutunda net ihtiyaçlara çeviren ve planın hayata geçirilmesi için bu ihtiyaçların giderilmesini planlayan stok yönetim model ve tekniklerini içine alır. Ana üretim planının uygulanması için lazım olan bütün parça ve malzemelerin ve bu gereksinimlerin karşılanması ile ilgili bilgiyi içeren ve MRP sistemi tarafından geliştirilen ana plan malzeme ihtiyaç planı olarak tanımlanır. Malzeme ihtiyaç planlaması sistemi uygulamalarında başarı seviyesinde;

- İşletmede insan kaynağının eğitim ve beceri düzeyi,
- İlişkili yan sistemlerin yeterliliği,
- Yönetimin desteği önemli derecede etkilidir.

Özkök'ün (1997), çalışmasına göre MRP, parça ve malzemelerin üretilmesi veya dışarıdan temin edilmesi gereken zamanı tespit eder. Ana düzeydeki malzemelerin ürün ağaçlarında bulunan tüm parçalar için bu planın yapılması gerekmektedir. Malzeme ihtiyaç planı ana düzey malzeme çizelgesinden hareketle oluşturulur. Bu çizelgede mamullere ne miktarda ve ne zaman ihtiyaç olduğu belirlenir. Sonrasında MRP ürün ağaçlarındaki parça ve malzemelerin temel bilgilerini inceleyerek lazım olan parçalar için planlanmış siparişleri oluşturur.

Yegül (2002), yaptığı çalışmaya göre MRP, ilk olarak 1960'ların başlarında ABD'de malzeme tedarikinde ve üretiminde bilgisayara dayalı bir yaklaşım olarak ortaya çıkmıştır. Bu tekniği tanımlayıcı kitap Orlicky tarafından 1975'te yayınlandı. Bu tekniğin ikinci dünya savaşı sonrasında Avrupa'da birkaç yerde bilgisayar olmaksızın kullanıldığı yönünde kayıtlar bulunmaktadır. Ancak Orlicky bu tekniğin bilgisayar kullanımıyla imalat stoklarını yönetmede daha detaylı uygulamaların yapılmasını sağladığını fark etmiştir.

MRP konusunda bilgisayarın etkin bir şekilde kullanımı Plossl ve Wight (1967) tarafından yapılmıştır. Plossl ve Wight MRP'nin hedefinde önemli yer teşkil eden;

- Verimli (az maliyetli) operasyonlar,
- Maksimum müşteri memnuniyeti,
- Minimum stok yatırımı hedeflerini yeniden tanımlamışlardır.

Yegül'ün (2002), yaptığı çalışmaya göre MRP'nin popülaritesi 1970'lerin başlarında Amerikan Üretim ve Stok Kontrol Topluluğu (APICS)'nın bu yöndeki teşvik edici uygulamalarıyla artmıştır. APICS, insanları MRP'nin tüm üretim prosesinin yönetiminde entegre iletişim ve karar destek sistemi olarak çözüm olduğu konusunda ikna etmeye çalışmıştır. Tekniğin optimize edilmesi için sistem analizinin ve yönetim biliminin gerekliliği üzerinde durulmuştur. En önemli sorunlar olarak disiplin, eğitim, anlayış ve iletişim olarak gösterilmiştir. Bu teşvik sonraları bilgisayar endüstri tarafından sürdürülmüştür.

3. MATERYAL VE METOT

3.1. Materyal

3.1.1. Uygulama sahası

Bu tezin uygulama kısmı, 2. Organize Sanayi Bölgesi Lalehan Caddesi No: 61 adresli, İmaş Makine Sanayi A.Ş.'de gerçekleştirilmiştir.

İmaş, 1989 yılında yüksek teknoloji ile profesyonel değirmen sistemleri ve testere tezgahları üretmek üzere kurulmuştur.

6600 m^2'si kapalı olmak üzere, 13.323 m^2'lik yerleşim alanına sahip tesislerinde ISO 9001 kalite sistemiyle üretim yapmaktadır.

İmaş, kurulduğu günden bu yana deneyimli teknik personeli ile un, irmik ve mısır unu fabrikaları kurmakta olup değirmen sektöründe iddialı bir kuruluştur.

Değirmen bina projesi, un ve irmik diyagram projelendirilmesi, servo-pnömatik vals, kare elek, elek kasaları, elektronik torbalama ve randıman kantarlarının üretimi başta olmak üzere tüm değirmen makinelerinin üretim, montaj ve devreye alınması (anahtar teslim projeler) dahil her türlü ihtiyaca cevap veren bir yapıyla çalışmaktadır.Bu noktada sadece Türkiye'nin değil, bölgenin ve dünyanın önde gelen, saygın üretici firmalarındandır.

İmaş, 2002 yılında geliştirdiği yeni dizayn servo-pnömatik valsleri kendi patentiyle üretim bandına eklemekle yetinmeyip, 2003 yılı içerisinde de sektöründe Türkiye'de bir ilke imza atarak değirmen fabrikalarının çok ihtiyaç duyduğu otomatik tavlama (nem ve akış kontrol) sistemini üreterek kullanıcıların hizmetine sunmuştur.

Üretilen makinelerde yerli hammadde, yarı mamul ve mamul kullanımına özen gösteren İmaş, üretimin %90'ını Afrika, Asya, Türk-i Cumhuriyetleri ve Avrupa ülkelerine ihraç etmektedir.

3.1.1.1 İmaş a.ş.'nin kalite politikası

İmaş'ın misyonu, koşulsuz müşteri memnuniyetinin sağlamak, müşterilerinin firmanın ürünlerine olan güvenini sürekli kılmak, rekabet gücünün sürekliliğini sağlamaktır.

İmaş, ortaklarının, çalışanlarının ve toplumun daha aydınlık bir geleceğe taşınması için;

- Müşterilerinin ve piyasaların istek ve beklentilerini sürekli izler,
- Ürünleriyle ilgili yasal ve teknik mevzuat şartlarına uygunluğu sağlar,
- Müşterilerine kaliteli ve üstün performanslı ürünler ile hızlı ve kaliteli servis desteği sunar,
- Kalite yönetim sisteminin etkinliğini sürekli iyileştirir,
- Çalışanların sürekli iyileştirme faaliyetlerine gönüllü katılımını sağlar,
- En önemli kaynağı olan çalışanlarının bilgi ve yeteneklerinin sürekli gelişmesini sağlar,
- Teknolojik yenilikleri takip eder.

İmaş, çevrenin korunmasının toplumumuza ve insanlığa karşı önemli bir sorumluluğu olduğu bilinciyle çalışır.

3.1.2. Bilgisayar desteği ve paket program

Algoritmaların çözüm sürecinde, formülasyonlar Microsoft Visual Studio 6.0 programında işletilmiştir. Donanım olarak da Intel Core 2 Duo İşlemci, 2.00 GHz, 2 GB RAM özelliğine ve Windows Vista Home Basic, Versiyon 2007, Service Pack 1 sistemine sahip bilgisayar kullanılmıştır.

3.2. Metot

Bu tezde izlenen yöntemde Konya'da faaliyet gösteren değirmen tezgahı üreticisi bir işletmede örnek olay çalışması gerçekleştirilmiştir. Malzeme ihtiyaç planlama sürecinde, parti-hacimlendirme probleminin çözümlenmesinde karar destek sistemi olarak Microsoft Visual Studio 6.0 yazılımı kullanılmış ve işletmedeki Vals makinesi ürün olarak ele alınıp sonrasında sipariş miktarı hesaplama yöntemleri bu ürünün aylık talepleri dikkate alınarak uygulanmıştır.

En sonunda ise 10 parti-hacimlendirme metodu ile Vals Makinesi için sipariş miktarı ve maliyetler bulunmuş, bunlar Tablo 4.18'de karşılaştırılarak bu hesaplama yöntemlerinden işletmeye önerileni ortaya konulmuştur.

Tez çalışması birtakım kısıtlar altında yapılmıştır. Kullanılan 10 parti-hacimlendirme metodunda işletmede planlama aylık yapıldığından dönemlerde zaman dilimi olarak ay baz alınmıştır.

Çalışmada veri edinme yöntemi olarak uygulama kısmı için işletmedeki planlama ve satın alma departmanlarındaki yetkililerle yüz yüze görüşmeler yapılmış ve üretim tesisi gezilerek gerekli dokümanlar alınmıştır. Diğer kısımların oluşturulmasında ise on-line veritabanları, üniversite kütüphane yayınlarından faydalanılmıştır.

3.3. MRP Sistemi ve MRP Sürecinde Sipariş Miktarı Hesaplama Yöntemleri

3.3.1. MRP'nin faydaları ve sakıncaları

MRP sistemi çok geniş bir bilgi işlem yükü gerektirir. Bilgisayar olmadan yüzlerce, binlerce stok kalemi parçalama ve sipariş verme faaliyetlerinin hızla ve doğru olarak yapılması neredeyse olanaksızdır.

MRP'nin temel mantığı basit olmakla birlikte göz önüne alınan stok kalemlerinin sayısının artması ve ürün yapılarının karmaşıklaşması işlem sayısını artırmakta ve sistemin bilgisayarla işletilmesini gerekli kılmaktadır (Güneş ve Firuzan 1999). Malzeme ihtiyaç planlaması sisteminin sağladığı katkılar ve sakıncaları aşağıdaki sıralanmıştır:

- Malzeme ihtiyaç planlaması talebi bağımlı olan malzemenin sipariş planlamasında uygun ve etkin bir mekanizmadır,
- Malzeme yönetiminde genellikle itme özelliği hakimdir,
- Üretim planları hem mevcut talebe hem de satış tahminlerine dayanarak yapılabilir,
- Bu sistem hem sipariş için de stok için işletilebilir,
- Bütün malzeme sistemine uyan entegre bir programdır,
- Stoklar uygun seviyede veya çok az miktarlarda tutulabilir,
- Stok yokluğu halinde önlem alınabilir,
- Montaj tipi üretimde daha etkin olmasına rağmen kesikli parti üretiminde de uygulanabilir,
- Sipariş verme ve hazırlık maliyetlerinin az miktarlarda olduğu üretim çeşitlerinde daha etkilidir,
- Uygulama aşamasında mutlaka bilgisayar desteği olmalıdır,
- Ana üretim planı düzeltilerek talepteki değişimlere cevap verilebilir. Fakat kısa dönemli dalgalanmalara uygulanması kolay değildir,
- Planlı ve yönetimi kolay bir mekanizma olduğu için makine verimleri artırılabilir,
- Daha kesin teslim vaatleri verilebilir ve müşteri ilişkileri kuvvetlendirilebilir,
- Karmaşık ve büyük sistemlerde kurulumda eksiklikler olabilir ve uygulamada bir takım zorluklar ortaya çıkabilir,
- Planlanan siparişler ile üretim kapasiteleri arasındaki ilişki rahatça kontrol edilebilir (Yenersoy 1990).

3.3.2. MRP sisteminin girdileri ve çıktıları

3.3.2.1. MRP sisteminin girdileri

Malzeme ihtiyaç planlamasının üç ana girdisi vardır. Bunlar:

- Ana üretim planı
- Ürün ağacı verileri
- Stok kayıt verileri

Malzeme ihtiyaç planlama sistemi bu üç ana girdi elde edilmeden işletilemez. Bundan dolayıdır ki MRP sistemini kullanmak isteyen işletmelerin ilk başta bu üç girdiyi sağlaması gerekmektedir (Yenersoy 1990).

3.3.2.1.1. Ana üretim planı

Ana üretim programı, belirli bir aydaki üretilecek bitmiş ürün miktarını içermektedir ve satış programına dayanılarak hazırlanmaktadır. Bunun satış programından temel farkı ise stoklardaki bitmiş parçalara göre ana üretim programının ayarlanmasıdır. Ayrıca yedek parça olarak üretimi yapılacak orijinal parçalar veya özel amaçlı deneyler için üretilecek ürünlerin miktarları da MRP sistemine girmektedir (Çelikçapa 2000).

Planlama işlemi her stok malzemesinin stok dengesini son üründen başlayarak ürün ağacının en alt kalemine kadar oluşturur. Eğer ihtiyaç, açık sipariş ve iş emirlerine bakılarak, stokta mevcut olan ve stoka girmesi planlanan miktarların toplanmasından daha fazla ise MRP, bu malzeme için bir sipariş planı yapar. Planlı sipariş, MRP'nin ihtiyaç duyulan malzeme için gerekli olan tarihte stokta bulunacak şekilde oluşturduğu sipariş veya iş emri teklifidir. Planlı siparişin miktarı, stok değerini, en az ihtiyacı karşılayacak miktara ulaştırır (Özkök 1997).

Ana üretim planı, MRP sistemi sürecini işleten bir mekanizmadır. Bu plan, hangi ürünlerin, ne zaman ve hangi miktarlarda üretileceğini belirten detaylı bir listedir ve yapılan talep tahminleri ve satış siparişlerine dayanarak hazırlanır. Başka bir ifadeyle müşteri siparişleri ve talep tahminlerinin, iş gücünü, makineleri ve malzemeleri en etkin şekilde kullanarak nasıl karşılanacağını gösteren bir çizelgedir. MRP sisteminin çıktıları bu plana bağlı olduğundan, siparişlerdeki azalma, artma, iptal gibi değişikliklerin sürekli olarak plana dahil edilmesi gerekmektedir. Ana planın girdileri son ürüne olan talep ve siparişler olduğundan çıktıları da son ürün cinsindendir (Çetinkaya 1988).

3.3.2.1.2. Malzeme fişi (Malzeme-parça-dağıtım fişi)

Malzeme ihtiyaç planlama sistemi genellikle üretim ve montaj işlemlerinin yapıldığı kuruluşlara uygundur. Üretilen ürünün malzemelerinde, bükme, kesme, delme, tornalama gibi işlemler olmalıdır. Montajda ise, yarı mamuller veya parçalar son ürünü ortaya çıkarabilmek için bir araya getirilmelidir. Malzeme ihtiyaç planlama sisteminde "parça" terimi son ürünün dışındaki yarı mamulleri, hammaddeleri, malzemeleri, kuruluş içi ve dışı üretilen veya satın alınan her şeyi içerir. Malzeme ihtiyaç planlama sisteminde, yalnız montaj ve parça ilişkileri göz önüne alınır. Bu ilişkiler malzeme fişleri vasıtasıyla belirlenir (Acar 1999).

Malzeme fişi, bir ürünü veya alt montajı oluşturan tüm parça ve malzemeleri listesidir. Her parça ve malzeme için bir stok numarası ve tanımı verildikten sonra bir adet ürün veya alt montaj için bu parça veya malzemeden kaç birim gerektiği gösterilmeli ve normal olarak hangi kaynaktan sağlandığı belirtilmelidir. Ürün tasarımı yapıldığında, malzeme fişlerinin de hazırlanması gerekir. Bu fişler sayesinde üretim gerçekleştirilebilir. Üründe olacak tasarım değişiklikleri de anında malzeme fişlerine aktarılmalıdır (Acar 1999).

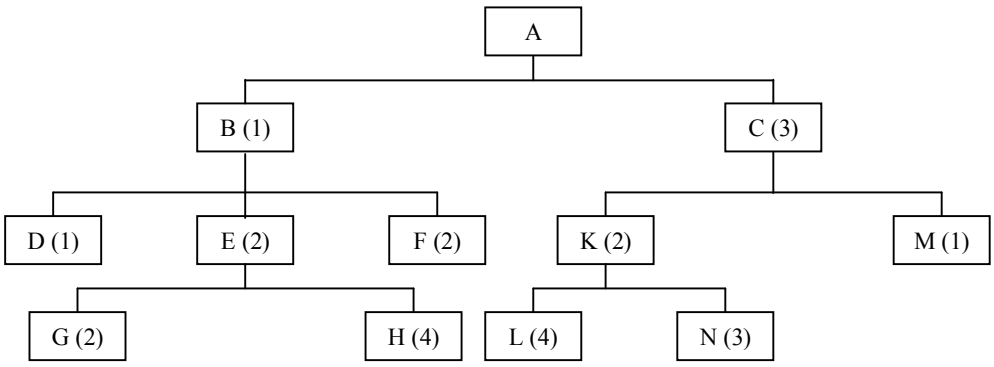
Malzeme ihtiyaç planlama sistemi, ürün bazında çalıştığından, malzeme fişleri büyük ölçüde kullanılır. Öte yandan malzeme fişlerinde olan bilgiler malzeme ihtiyaç planlama sisteminde olduğu gibi aktarılmaz, fişlerdeki bilgilerin uygun bir yapıya dönüştürülmesi gerekir. Bu yapıya ürün ağacı denir (Acar 1999).

3.3.2.1.3. Ürün ağacı

Malzeme ihtiyaç planlama sisteminde montaj ve parça ilişkileri dikkate alınır. Ürün ağacı, bitmiş ürünü üretebilmek için kullanılan bütün parça, yarı mamul, malzemeleri ve bunların miktarlarını tespit eder. Ürün ağaçlarının organizasyon yapısını anımsatan görüntüsünde son mamulden başlanarak, aşağıya doğru son ürünü oluşturan tüm malzeme ve ana parçaların miktarları, ürün isimleri ve parça numaraları verileri oluşturulur. Ayrıca ürün ağacı, bileşenlerin temin süreleri ve kademe kodları gibi verileri de sağlamaktadır (Acar 1991).

Ürün ağacı, hangi parçalardan hangi miktarda ve hangi seviyede yararlanılacağını göstermektedir. Malzeme ihtiyaç planlamasını etkileyen bir başka faktör de tek seferde üretilecek parçaların miktarı olmaktadır. Bu miktar parçaların taşınacağı kabın ölçüleri, ekonomik üretim miktarı gibi etkenlere göre belirlenmektedir (Acar 1991).

Ürünün bütün bileşenlerinin geriye doğru dökümünün sistematik çatısını oluşturmak amacıyla bir kodlama sistemi tasarlanmıştır. Bu sistemde nihai üründen başlayarak her ürün ağacına bir kademe kodu verilmektedir. Aşağıdaki örnekte verilen ürün ağacından faydalanılarak ihtiyaç duyulan miktarlar bulunmuştur (Acar 1991).



Şekil 3.1 A Ürünü İçin Kademe Kodlaması (Acar 1991)

Şekil 3.1'de gösterildiği gibi 1 adet A ürünü meydana getirebilmek için 1 tane M ve 3 tane C gereklidir. Aynı şekilde 1 adet B üretmek için 1 adet D, 2 adet E ve 2 adet F gerekmektedir. 1 adet C ürünü üretebilmek için 2 adet K ve 1 adet E gereklidir. 1 adet E üretebilmek için 2 adet G, 1 adet H gereklidir. 1 adet K üretmek için 4 adet L ve 3 adet N gereklidir (Acar 1991).

10 adet A ürününden üretmek için;

B: 1*10 = 10 adet

C: 3*10 = 30 adet

D: 1*10 = 10 adet

E: 2*10 = 20 adet

F: 2*10 = 20 adet

K: 2*30 = 60 adet

M: 1*30 = 30 adet

G: 2*20 = 40 adet

H: 4*20 = 80 adet

L: 4*60 = 240 adet

N: 3*60 = 180 adet

3.3.2.1.4. Stok durumu bilgileri

Stok kayıtları, eldeki mevcut ürünlerin, yarı mamullerin ve parçaların miktarını içermektedir. Ana üretim programı ve stok miktarına göre üretilecek parçaların önceliği de belli olmaktadır (Çelikçapa 2000).

Stok bilgileri genelde depodaki bütün malzemeler için, malzeme giriş, çıkış, sipariş, temin süresi, temin yeri, sipariş miktarları gibi verilerin tutulduğu bir kayıt setidir. Bu kayıtlar, her bir stok kaleminin durumu hakkındaki verileri saklamak için kaydedilmektedir. Ürünün yapısal şemasında tanımlanan ve siparişi planlanacak olan her çeşit montaj elemanın envanter ve sipariş durumları ile ilgili türlü verileri içerir. Bu veriler parçanın kod numarası, tanımı, emniyet stoku gibi bilgilerdir (Güneş ve Firuzan 1999).

3.3.2.2. MRP sisteminin çıktıları

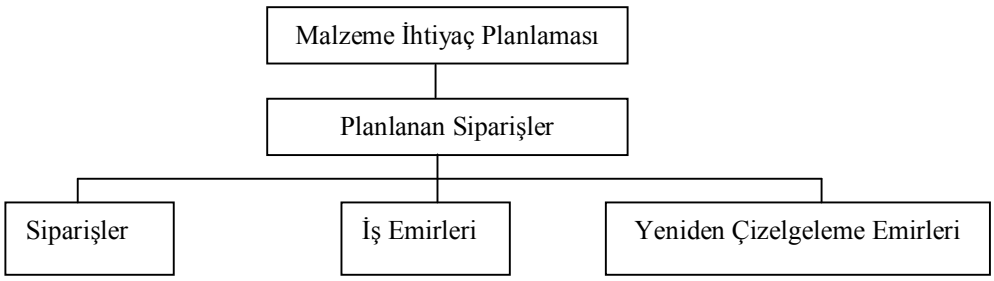
Malzeme ihtiyaç planlaması sisteminin çıktıları aşağıdaki gibi özetlenebilir.

- Sipariş açma ikazları – planlanan sipariş açımı için,
- Yeniden çizelgeleme ikazları – daha önceden açılmış olan siparişlerin teslim tarihlerinde bir değişiklik olduğunda gerekli düzeltmelerin gerçekleştirilmesi için,
- İptal etme ikazları – daha önceden açılmış siparişlerin kapatılması veya daha ileri bir tarihe atılması için,
- İleriki dönemde açılmak üzere çizelgelenmiş siparişler.

Bu ana çıktıların yanında MRP sisteminden, kullanıcılar daha farklı raporlar da isteyebilir ve üretebilirler. Bunalrdan bazıları aşağıda verilmiştir.

- Stok seviyesi izdüşümü (envanter tahminleri)
- Performans analizleri
- Talep kaynakları araştırma raporları

Yukarıda da görüldüğü gibi MRP sistemi çıktıları çok yönlüdür ve yöneticiler istekleri doğrultusunda bunlar çeşitlendirilebilir. Fakat standar malzeme ihtiyaç planlama sistemindeki ana çıktılar Şekil 3.2'de gösterildiği gibidir (Acar 1999).



Şekil 3.2. MRP'nin Çıktıları (Dağlı 1987)

3.3.2.2.1. Sipariş önceliklerinin planlama raporları

Sipariş önceliklerinin planlama raporları gerçek ihtiyaç zamanları ile açılan sipariş teslim tarihlerini kıyaslayarak, ortaya çıkabilecek sapma ve gecikmeleri önceden haber vermektedir. Bu çıktıların hazırlanmasında MRP sistemi, etkilenen birimlerin hangi doğrultuda ve ne kadarlık bir zaman dilimi için çizelgeleneceğini belirtir.

MRP, sipariş önceliklerini gerçekçi bir şekilde korumak ve birimlerin stok durumları ile ilgili problemleri ana üretim planı ile ilişkilendirmek durumundadır. Önceliklerin gerçekçi olması için, ana üretim planı, kapasite, malzeme ve temin süresi açısından karşılanması imkansız olan son ürün gereksinimlerini içermelidir (Acar 1991).

3.3.2.2.2. Performans kontrol raporları

Bu tip raporlar, malzeme ihtiyaç planlaması sisteminin yan çıktıları olup, yönetimin, stok programcılarının, müşterilerin ve tedarikçilerin parçaları denetlemesini sağlamaktadır.

3.3.2.2.3. Temin süreleri

B ve C bileşenlerinin bir imalat operasyonuyla birleştirilip A alt montajının üretildiğini varsayalım. Bu işlem belirli bir zaman süresinde tamamlanacaktır. Eğer A parçasının belirli bir tarihte hazır olması gerekiyorsa söz konusu imalat operasyonunun başlaması için uygun bir tarih tespit edilmelidir. Diğer bir değişle, B ve C bileşenlerinin bu tarihlerde hazır olmaları gerekir (Acar 1999).

Sipariş miktarlarının tespiti daha çok ortaya koyulacak sipariş parti büyüklüğü politikası ile ilgilidir.

Parti büyüklüğünü belirlemede fiyat ıskontolarından faydalanma, birim miktar başına hazırlık maliyetlerini düşük tutma, satın alma faaliyetlerinin rahatça takibine imkan sağlayacak miktarda siparişi açma gibi parametreleri göz önünde bulundurmak gerekmektedir (Hamarat 1986).

Mamul üretiminde yararlanılan parçaların talebi mamulün talebine bağımlıdır. Parça taleplerinin bağımlı olmasından ve üretimin seri olmasından dolayı kesikli talep oluşmakta ve geleneksel envanter kontrol metotlarının kullanılması mümkün olmamaktadır. Ekonomik sipariş miktarı metotları yalnızca en alt düzeydeki parça ve malzemeler için uygun olmaktadır. Eğer bu metottaki hesaplamalar yukarı düzeylere uygulanırsa alt düzeydekilere olan talebi yoğunlaştıracaktır. Onun için MRP için en uygun sipariş miktarını hesaplayan bir takım metotlar geliştirilmiştir (Hamarat 1986).

Bu metotlardan en fazla uygulananları aşağıda belirtilmiştir.

3.3.3. MRP sürecinde sipariş miktarı hesaplama yöntemleri

3.3.3.1. Sabit sipariş miktarı yöntemi

Bu metoda göre bir parçanın bütün siparişleri eşit büyüklüklerde planlanır. Genelde parçaya olan talep düzenli ise tercih edilmektedir. Sabit sipariş miktarı yöntemi, Tablo 3.1'de ele alınmıştır. Fakat metodun uygulanabilirliği talebi düzenli olan parçaların varlığı ile sınırlandırılmıştır (Hamarat 1986). (stok taşıma maliyeti dönem başına 1 para birimi ve tedarik süresi 1 ay olarak alınmıştır.)

Tablo 3.1. Sabit Sipariş Miktarı Yöntemi (Hamarat 1986)

Dönem	1	2	3	4	5	6	7	8	9	10	11	12	Toplam
Net İhtiyaçlar	55	20	40	35	25	40	55	60	70	80	70	50	600
Verilen Sipariş	100		100			100		100		100	100		600
Stok Taşıma Maliyeti	0	45	25	85	50	25	85	30	70	0	20	50	485
Hazırlık Maliyeti	100		100			100		100		100	100		600
Toplam Maliyet													1085

3.3.3.2. Net ihtiyaç kadar sipariş yöntemi

Tablo 3.2'de görüldüğü üzere, verilen sipariş miktarı net ihtiyaç miktarına eşit olmaktadır. Net ihtiyaçtan az yada fazla miktarda sipariş verilmez. Her dönemin net ihtiyacı farklı bir sipariş iş emriyle girilir. Bu metot maliyeti fazla olan malzemeler için uygundur. Burada hedef envanter maliyetini minimize etmektir (Hamarat 1986).

Tablo 3.2. Net İhtiyaç Kadar Sipariş Yöntemi (Hamarat 1986)

Dönem		1	2	3	4	5	6	7	8	9	10	11	12	Toplam
Net İhtiyaçlar		55	20	40	35	25	40	55	60	70	80	70	50	600
Verilen Sipariş	55	20	40	35	25	40	55	60	70	80	70	50	55	600
Stok Taşıma Maliyeti	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Hazırlık Maliyeti	100	100	100	100	100	100	100	100	100	100	100	100	100	1200
Toplam Maliyet														1085

3.3.3.3. Ekonomik sipariş miktarı yöntemi

Ekonomik sipariş miktarı (EOQ) yöntemi, MRP sisteminde kullanılmak amacıyla geliştirilmiş bir politikadır. Bu yöntemde, siparişlerin verilme zamanı belirli bir periyoda bağımlı değildir. Dönemler talebin durumuna göre net ihtiyacı karşılayacak şekilde ayarlanmaktadır. Tablo 3.3'e bakıldığında eğer sipariş miktarı dönem talebini karşılamaya yetmiyorsa ya dönem sayısı azaltılır yada talep karşılanıncaya kadar miktar artırılır (Hamarat 1986).

Ekonomik sipariş miktarı, genellikle düşük düzeydeki ihtiyaçlar, hammaddeler ve aynı parçaya bağlı bileşenler için talebin düzgün ve sürekli olduğu durumlarda iyi neticeler verir. Eşitlik 3.1'de hesaplanışı gösterilmiştir (Hamarat 1986).

Hazırlık maliyeti (S): 100 YTL

Stok taşıma maliyeti (I): 12 YTL yıllık

Yıllık kullanım miktarı (U): 600

$$EOQ = \sqrt{(2*U*S)/(I)} = \sqrt{(2*600*100)/(12)} = 100 \qquad (3.1.)$$

Tablo 3.3. Ekonomik Sipariş Miktarı Yöntemi (Hamarat 1986)

Dönem	1	2	3	4	5	6	7	8	9	10	11	12	Toplam
Net İhtiyaçlar	55	20	40	35	25	40	55	60	70	80	70	50	600
Verilen Sipariş	100		100			100				100	100		600
Stok Taşıma Maliyeti	0	45	25	85	50	25	85	30	70	0	20	50	485
Hazırlık Maliyeti	100		100			100		100		100	100		600
Toplam Maliyet													1085

3.3.3.4. Sabit dönem algoritması

Bu metotta siparişler belli bir dönemin, mesela iki dönemlik sipariş verme aralığı belirlenerek gereksinim duyulan miktarlar toplanır ve siparişler verilir. Geçmişteki satın alma ve talep zamanına bakılarak bu süreye karar verilebilir. Tablo 3.4'teki gibi genellikle bir siparişte iki dönemin siparişi verilir (Hamarat 1986).

Tablo 3.4. Sabit Dönem Algoritması (Hamarat 1986)

Dönem	1	2	3	4	5	6	7	8	9	10	11	12	Toplam
Net İhtiyaçlar	55	20	40	35	25	40	55	60	70	80	70	50	600
Verilen Sipariş	115			100			185			200			600
Stok Taşıma Maliyeti	0	60	40	0	65	40	0	130	70	0	120	50	575
Hazırlık Maliyeti	100			100			100			100			400
Toplam Maliyet													975

3.3.3.5. Periyodik sipariş miktarı yöntemi

Geleneksel ekonomik sipariş miktarı yaklaşımından, kesikli dönemsel talep ortamında yararlanılmak üzere geliştirilmesi ile bu metot elde edilmiştir.

Ekonomik sipariş miktarı örneğinde EOQ = 100 olarak bulunmuştu.

Bir yıllık dönem sayısı = 12

Yıllık talep = 600

600/100 = 6 (Yıllık sipariş sayısı)

12/6 = 2 (Sipariş verme aralığı)

Bu sonuçların değerlendirilmesi Tablo 3.5'te verilmiştir. Burada sipariş verme aralığının iki yada üç dönem arasında değiştiği varsayılmıştır. Ancak bu metot, talebin kesikli ve düzgün olmadığı durumlarda zayıf kalabilmektedir (Acar 1991).

Tablo 3.5. Periyodik Sipariş Miktarı (Acar 1991)

Dönem	1	2	3	4	5	6	7	8	9	10	11	12	Toplam
Net İhtiyaçlar	55	20	40	35	25	40	55	60	70	80	70	50	600
Verilen Sipariş	75		75		65		115		150		120		600
Stok Taşıma Maliyeti	0	20	0	35	0	0	0	60	0	80	0	50	295
Hazırlık Maliyeti	100		100		100		100				100		600
Toplam Maliyet													895

3.3.3.6. En düşük birim maliyet yöntemi

Tablo 3.6 ve Tablo 3.7'de gösterildiği gibi bu metotta, sipariş miktarının sadece ilk dönem net ihtiyaçlarını veya bir sonraki dönem veya ondan sonraki dönemlerin de net ihtiyaçlarını karşılayıp karşılayamadığına test edilir (birim başına düşen hazırlık maliyeti + stok taşıma maliyeti) ve incelenir. Bu maliyeti minimize eden miktar, sipariş miktarı olarak tayin edilir (Acar 1991).

Tablo 3.6. En Düşük Birim Maliyetin Hesaplanması (Acar 1991)

Dönem	Net İhtiyaç	Stokta Taşındığı Dönem Sayısı	Muhtemel Sipariş Miktarı (1)	Taşıma Parti İçin (2)	Maliyet Birim İçin (2/1)	Birim Hazırlık Maliyeti (100/1)	Birim Maliyet
1	55	0	55	0	0	1,81	1,81
2	20	1	75 *	20	0,26	1,33	1,59
3	40	2	115	100	0,86	0,87	1,73
1	40	0	40	0	0	2,5	2,5
2	35	1	75 *	35	0,46	1,33	1,79
3	25	2	100	85	0,85	1	1,85
1	25	0	25	0	0	4	4
2	40	1	65	40	0,61	1,54	2,15
3	55	2	120 *	150	1,25	0,83	2,08
4	60	3	180	330	1,83	0,55	2,38
1	60	0	60	0	0	1,67	1,67
2	70	1	130 *	70	0,54	0,77	1,31
3	80	2	210	250	1,19	0,47	1,66
1	80	0	80	0	0	1,25	1,25
2	70	1	150 *	70	0,46	0,67	1,13
3	50	2	200	170	0,85	0,5	1,35
1	50	0	50 *	0	0	2	2

Tablo 3.7. En Düşük Birim Maliyet Sipariş Tablosu (Acar 1991)

Dönem	1	2	3	4	5	6	7	8	9	10	11	12	Toplam
Net İhtiyaçlar	55	20	40	35	25	40	55	60	70	80	70	50	600
Verilen Sipariş	75		75		120			130		150		50	600
Stok Taşıma Maliyeti	0	20	0	35	0	95	55	0	0	0	70	0	345
Hazırlık Maliyeti	100		100		100			100		100		100	600
Toplam Maliyet													945

3.3.3.7. En düşük toplam maliyet yöntemi

Bu yaklaşımda kullanılan varsayım; planlama dönemindeki bütün partiler için hazırlık ve stok taşıma maliyetleri toplamının asgari miktara düşürülmesi için, partilerin toplam maliyetlerinin birbirlerine eşit olması gerekmektedir.

En düşük toplam maliyet yaklaşımı, bu hedefe erişmek için, birim başına hazırlık maliyeti ile stok taşıma maliyetinin eşit olduğu miktarlarda sipariş vermektedir. Bu yöntem, maliyetlerin eşitliğini sağlamak için ekonomik parça-dönem faktörü (EPP) olarak tanımlanan bir kavramdan faydalanır. Ekonomik parça-dönem faktörü, stokta bir dönem taşındığında, hazırlık maliyetine eşit miktardaki taşıma maliyetini verecek olan birim miktar olarak tanımlanmaktadır (Acar 1991).

Şöyleki;

EPP = En düşük toplam maliyet

S = Hazırlık maliyeti

Ip = Dönemsel stok taşıma maliyeti

C = Birim maliyet

$$EPP = S/(Ip*C) = 100/(0,02*50) = 100 \qquad (3.2.)$$

En düşük toplam maliyet yaklaşımı, parça-dönem maliyetinin EPP değerine en yakın olduğu sipariş miktarını seçmektedir. Bu hesaplama Tablo 3.8'de, sipariş tablosu ise Tablo 3.9'da verilmiş ve eşitlik 3.2'de hesaplanışı gösterilmiştir.

Tablo 3.8. En Düşük Toplam Maliyetin Hesaplanması (Acar 1991)

Dönem	Net İhtiyaç	Stokta Taşındığı Dönem Sayısı	Muhtemel Parti Büyüklüğü	Parça-Dönem (Kümülatif)
1	55	0	55	0
2	20	1	75	20
3	40	2	115 *	100
4	35	3	150	205
1	35	0	35	0
2	25	1	60 *	25
3	40	2	100	105
1	40	0	40	0
2	55	1	95 *	55
3	60	2	155	175
1	60	0	60	0
2	70	1	130 *	70
3	80	2	210	230
1	80	0	80	0
2	70	1	150 *	70
3	50	2	200	170
1	50	0	50 *	0

Tablo 3.9. En düşük Toplam Maliyet Sipariş Tablosu (Acar 1991)

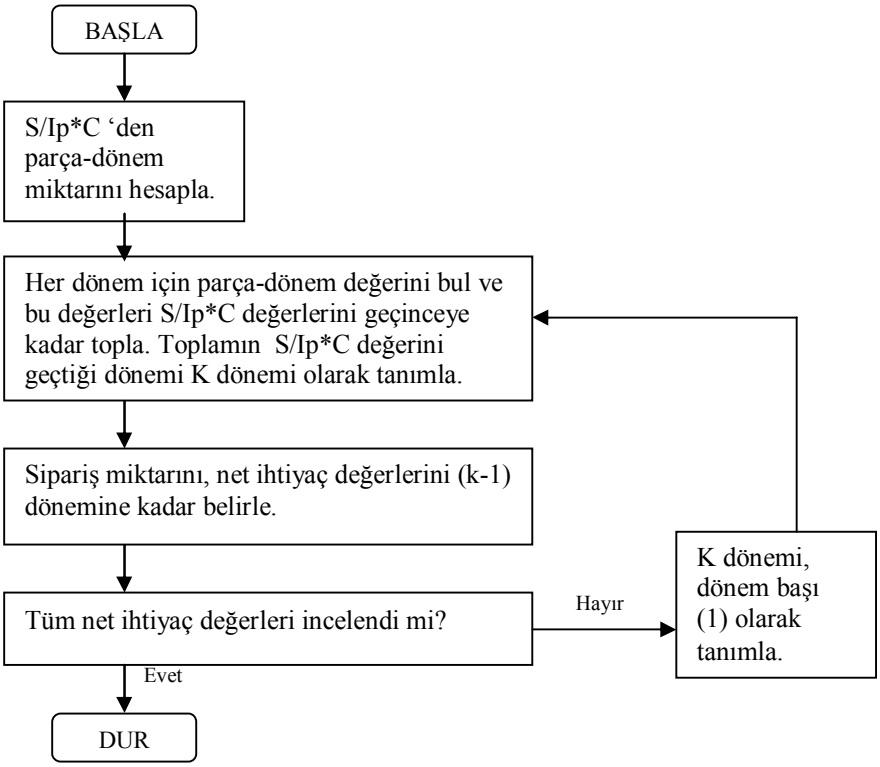
Dönem	1	2	3	4	5	6	7	8	9	10	11	12	Toplam
Net İhtiyaçlar	55	20	40	35	25	40	55	60	70	80	70	50	600
Verilen Sipariş	115			60		95		130		150		50	600
Stok Taşıma Maliyeti	0	60	40	0	25	0	55	0	70	0	70	0	320
Hazırlık Maliyeti	100			100		100		100		100		100	600
Toplam Maliyet													920

3.3.3.8. Parça dönem algoritması

Sipariş miktarlarının hesaplanmasında yararlanılan diğer bir metot ise parça-dönem algoritmasıdır. Bu algoritmanın temel mantığı sipariş verme maliyetleri ile envanter taşıma maliyetlerini zaman faktörü altında kıyaslayarak sipariş miktarlarını bu malzemelerin değişimine göre tayin etmektir. Toplam sipariş verme maliyetinin toplam envanter taşıma maliyetine eşitlendiği noktada sipariş miktarı tespit edilmektedir.

Bu algoritma, temelde en düşük toplam maliyet metodu ile çok büyük benzerlikler göstermektedir. Fakat, burada sipariş miktarları ve zamanları daha farklı bir şekilde belirlenmektedir.

Parça-dönem algoritması, sipariş verme zamanı belli bir zaman içinde birim zamanda taşınan stokun EPP (En düşük toplam maliyet) miktarını aştığı zamandır. Sipariş miktarı ise bu döneme kadar olan taleptir. EPP değerini aştığı dönemdeki talep dahil edilmez (Acar 1991). Şekil 3.3'te algoritma, Tablo 3.10'da ise bu algoritmaya ait sipariş tablosu gösterilmiştir.



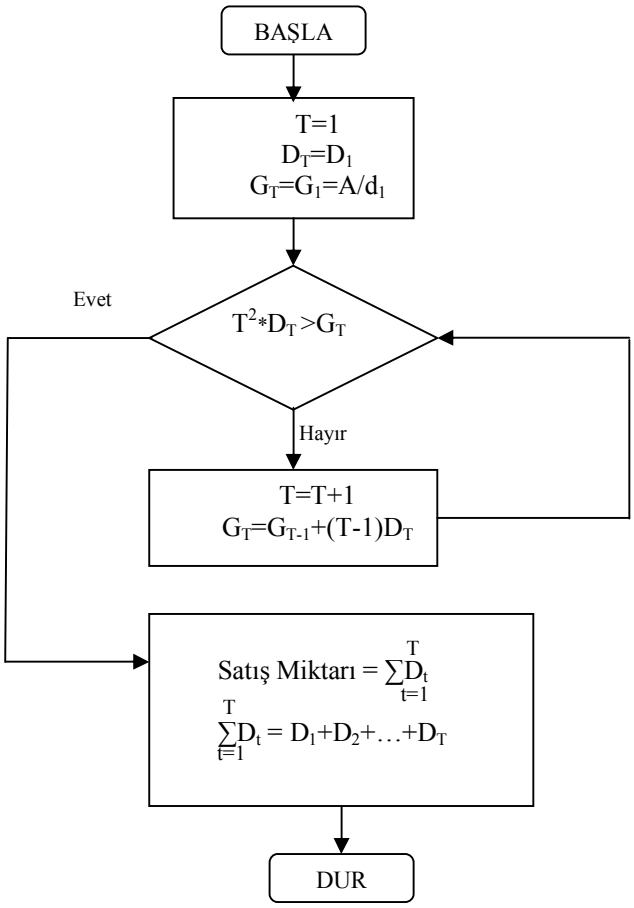
Şekil 3.3. Parça-Dönem Algoritması (Acar 1991)

Tablo 3.10. Parça-Dönem Algoritması Sipariş Tablosu (Acar 1991)

Dönem	1	2	3	4	5	6	7	8	9	10	11	12	Toplam
Net İhtiyaçlar	55	20	40	35	25	40	55	60	70	80	70	50	600
Parça-Dönem	0	1* 20	2* 40	3* 35	1* 25	2* 25	1* 40	2* 60	1* 70	2* 80	1* 70	2* 50	600
Parça-Dönem (Kümülatif)	0	20	100	205	25	105	55	175	70	230	70	170	320
Sipariş Miktarı	115			60		95		130		150		50	600
Dönem	1	2	3	4	5	6	7	8	9	10	11	12	Toplam
Net İhtiyaçlar	55	20	40	35	25	40	55	60	70	80	70	50	600
Verilen Sipariş	115			60		95		130		150		50	600
Stok Taşıma Maliyeti	0	60	40	0	25	0	55	0	70	0	70	0	320
Hazırlık Maliyeti	100			100		100		100		100		100	600
Toplam Maliyet													920

3.3.3.9. Silver-Meal sezgisel algoritması

Silver-Meal algoritması talebin dönemlere göre değiştiğini varsaymaktadır. Birim zamana düşen toplam maliyet miktarını en aza indirerek sipariş verilecek olan miktarı belirlemeye çalışan ve optimuma yakın bir çözüm veren sezgisel bir algoritmadır. Algoritmanın temel yapısını açıklamak için Şekil 3.4'ten sonraki tanımlamalara yer verilmiştir. Tablo 3.11'de ise bu algoritmaya ait sipariş tablosu yer almaktadır.



Şekil 3.4. Silver-Meal Sezgisel Algoritması (Hamarat 1986)

D_T : t dönemindeki talep

T : Siparişin talebi karşılayacağı süre (dönem)

A : Hazırlık maliyeti

h : Envanter taşıma maliyeti

c : Birim maliyeti

G_T : Talebin Birimkimli Dağılım Fonksiyonu

$G_T = G_{T-1} + (T-1)*D_T$ (3.3.)

T = 1 için $D_T = D_1 = 55$

$G_T = G_1 = 100/0{,}02*50 = 100 > 55$

T = 2 için $G_T = G_{T-1} + (T-1)*D_T = 100 + 1*20 = 120$

$T^2 * D_T = 4*20 = 80<120$

T =3 için $G_T = 120 + 2*40 = 200$

$T^2 * D_T = 9*40 = 360 > 200$ olduğundan sipariş miktarı:

$$\sum_{t=1}^{2} D_T = 55+20 = 75$$

T = 1 için $D_T = D_1 = 40$

$G_T = G_1 = 100/0{,}02*50 = 100 > 40$

T = 2 için $G_T = G_{T-1} + (T-1)*D_T = 100 + 1*35 = 120$

$T^2 * D_T = 4*35 = 140>35$ olduğundan sipariş miktarı:

$$\sum_{t=1}^{1} D_T = 40$$

T = 1 için $D_T = D_1 = 35$

$G_T = G_1 = 100/0{,}02*50 = 100 > 35$

T = 2 için $G_T = G_{T-1} + (T-1)*D_T = 100 + 1*25 = 125$

$T^2 * D_T = 4*25 = 100<125$

T =3 için $G_T = 100 + 2*40 = 180$

$T^2 * D_T = 9*40 = 360>180$ olduğundan sipariş miktarı

$$\sum_{t=1}^{2} D_T = 35+25 = 60$$

T = 1 için $D_T = D_1 = 40$

$G_T = G_1 = 100/0{,}02*50 = 100 > 40$

T = 2 için $G_T = G_{T-1} + (T-1)*D_T = 100 + 1*55 = 155$

$T^2 * D_T = 4*55 = 220>155$ olduğundan sipariş miktarı:

$\sum_{t=1}^{1} D_T = 40$

T = 1 için $D_T = D_1 = 55$

$G_T = G_1 = 100/0{,}02*50 = 100 > 55$

T = 2 için $G_T = G_{T-1} + (T-1)*D_T = 100 + 1*60 = 160$

$T^2 * D_T = 4*60 = 240>160$ olduğundan sipariş miktarı:

$\sum_{t=1}^{1} D_T = 55$

T = 1 için $D_T = D_1 = 60$

$G_T = G_1 = 100/0{,}02*50 = 100 > 60$

T = 2 için $G_T = G_{T-1} + (T-1)*D_T = 100 + 1*70 = 170$

$T^2 *D_T = 4*70 = 280>170$ olduğundan sipariş miktarı:

$\sum_{t=1}^{1} D_T = 60$

T = 1 için $D_T = D_1 = 80$

$G_T = G_1 = 100/0{,}02*50 = 100 > 80$

T = 2 için $G_T = G_{T-1} + (T-1)*D_T = 100 + 1*70 = 170$

$T^2 * D_T = 4*70 = 280>170$ olduğundan sipariş miktarı:

$\sum_{t=1}^{1} D_T = 80$

T = 1 için $D_T = D_1 = 70$

$G_T = G_1 = 100/0{,}02*50 = 100 > 70$

T = 2 için $G_T = G_{T-1} + (T-1)*D_T = 100 + 1*50 = 150$

$T^2 *D_T = 4*50 =200>150$ olduğundan sipariş miktarı:

$\sum_{t=1}^{1} D_T = 70$

Tablo 3.11. Silver-Meal Algoritması Sipariş Tablosu (Hamarat 1986)

Dönem	1	2	3	4	5	6	7	8	9	10	11	12	Toplam
Net İhtiyaçlar	55	20	40	35	25	40	55	60	70	80	70	50	600
Verilen Sipariş	75		40	60		40	55	60	70	80	70	50	600
Stok Taşıma Maliyeti	0	20	0	0	25	0	0	0	0	0	0	0	45
Hazırlık Maliyeti	100		100	100		100	100	100	100	100	100	100	1000
Toplam Maliyet													1045

3.3.3.10. Wagner-Whitin algoritması

Bu algoritma dinamik programlama tekniğinin kullanıldığı bir optimizasyon yöntemidir. Temelde en uygun planlı siparişleri elde etmek için bütün mümkün yollar değerlendirilmektedir. Bu değerlendirme, planlama döneminde yer alan bütün periyotlar için birlikte dikkate alınarak yapılmaktadır. Algoritma sipariş verme ve elde bulundurma maliyetlerini minimize ederek en uygun sonucu verir. Bu algoritma ile ulaşılan değerler diğer tekniklerin etkinliğinin belirlenmesinde bir ölçü olabilir.

Wagner-Whitin algoritması teoride optimum çözüm veren olarak kabul edilmesine karşın binlerce kalemi bulan envanter grubu için pratikteki yapılabilirliği sınırlı kalmaktadır. Çünkü algoritmanın karmaşık hesaplama yapısı bilgisayarlardan yararlanılmasına rağmen hesaplama süresinin çok fazla uzamasına neden olmaktadır (Hamarat 1986).

Bu metot, dinamik kökenli bir optimizasyon tekniğini kullanır. Bu metot, her periyottaki taleplere cevap verebilecek şekilde sipariş verme yollarının her birini dener ve verilen net ihtiyaç çizelgesi için optimum (en iyi) sipariş verme politikasını belirler. Wagner-Whitin algoritması optimaldir, diğer teknikler ise sezgiseldir. Tablo 3.12. a.'da stok maliyetlerinin hesaplanması, Tablo 3.12. b'de ise toplam stok maliyetlerinin hesaplanması verilirken, Tablo 3.12. c.'de siparişlerin belirlenmesi ve Tablo 3.12. d'de ise sipariş tablosu gösterilmiştir.

Tablo 3.12. a. Wagner-Whitin Algoritması Stok Maliyetlerinin Hesaplanması (Hamarat 1986)

Dönem	1	2	3	4	5	6	7	8	9	10	11	12
Talep	55	20	40	35	25	40	55	60	70	80	70	50
Hiz. Sip. Mal.	100	100	100	100	100	100	100	100	100	100	100	100
Stok Taşıma Maliyeti	1	1	1	1	1	1	1	1	1	1	1	1
Stokta Taşınan Dönem Sayısı	0	1	2	3	4	5	6	7	8	9	10	11
	100	20	80	105	100	200	330	420	560	720	700	550
		100	40	70	75	160	275	360	490	640	630	500
			100	35	50	120	220	300	420	560	540	450
				100	25	80	165	240	350	480	490	400
					100	40	110	180	280	400	420	350
						100	55	120	210	320	350	300
							100	60	140	240	280	250
								100	70	160	210	200
									100	80	140	150
										100	70	100
											100	50
												100

Tablo 3.12. b. Wagner-Whitin Algoritması Toplam Stok Maliyetlerinin Hesaplanması (Hamarat 1986)

Dönem	1	2	3	4	5	6	7	8	9	10	11	12
1	100	120	200	305	405	605	935	1355	1915	2635	3335	3885
2		100	140	210	285	445	720	1080	1570	2210	2840	3340
3			100	135	185	305	525	825	1245	1805	2345	2795
4				100	125	205	370	610	960	1440	1930	2330
5					100	140	275	430	710	1110	1530	1880
6						100	155	275	485	805	1155	1455
7							100	160	300	540	820	1070
8								100	170	320	540	740
9									100	180	320	470
10										100	170	270
11											100	150
12												100

Tablo 3.12. c. Wagner-Whitin Algoritması Uygun Siparişlerin Belirlenmesi (Hamarat 1986)

Dönem		1	2	3	4	5	6	7	8	9	10	11	12
0	1	100	120	200	305	405	605	935	1355	1915	2635	3335	3885
100	2		100	140	210	285	445	720	1080	1570	2210	2840	3340
120	3			100	135	185	305	525	825	1245	1805	2345	2795
200	4				100	125	205	370	610	960	1440	1930	2330
305	5					100	140	275	430	710	1110	1530	1880
405	6						100	155	275	485	805	1155	1455
605	7							100	160	300	540	820	1070
935	8								100	170	320	540	740
1355	9									100	180	320	470
1915	10										100	170	270
2635	11											100	150
3335	12												100

Tablo 3.12. d. Wagner-Whitin Algoritması Sipariş Tablosu (Hamarat 1986)

Dönem	1	2	3	4	5	6	7	8	9	10	11	12	Toplam
Net İhtiyaçlar	55	20	40	35	25	40	55	60	70	80	70	50	600
Verilen Sipariş	115			100			385						600
Stok Taşıma Maliyeti	0	60	40	0	65	40	0	330	270	200	120	50	1175
Hazırlık Maliyeti	100			100			100						300
Toplam Maliyet													1475

Aşağıda verilen tabloda şimdiye kadar yapılan malzeme ihtiyaç planlama sistemine ait hesaplamaların sonuçları gösterilmiş ve çözülen örnek için minimum maliyeti veren sonuç altı çizili olarak belirtilmiştir.

Tablo 3.13. MRP Sipariş Hesaplamaların Karşılaştırılması

Sipariş Miktarı Hesaplama Algoritmaları	Sonuçlar
Sabit sipariş miktarı	1085
Net ihtiyaç kadar sipariş	1200
Ekonomik sipariş miktarı	1085
Sabit dönem algoritması	975
Periyodik sipariş miktarı	**895**
En düşük birim maliyet	945
En düşük toplam maliyet	920
Parça dönem algoritması	920
Silver-Meal sezgisel	1045
Wagner-Whitin algoritması	1475

3.4. Karar Destek Sistemleri

Bazen karar verici, kaliteli bir karar vermek için tecrübesine güvenebilir veya Yönetim Bilişim Sistemi'nden (YBS) elde edilen mevcut bilgiden başka bir bilgiye bakmaya ihtiyaç duymaz. Özellikle taktik ve stratejik düzeylerdeki karar vericiler, sıklıkla karmaşık etkenlerin tam olarak sentez edilmesi insan yeteneğinin üstünde olan karmaşık kararlarla karşı karşıya kalırlar. Bu tip kararlar karar destek sistemleri (KDS) için uygundur (Long 1989, Yozgat 1998). KDS, YBS'lerden daha esnektir ve farklı durumlar için karar vericiye yardım desteği sunabilmektedir. Tüm karar aşamaları, karar tipleri ve farklı yapıdaki problemlerle ilgilenebilir (Stair 1992, Yozgat 1998).

Karar destek sistemleri ile ilgili yapılan değişik birkaç tanım aşağıda verilmiştir:

- Bir karar Destek Sistemi, kullanıcıya yarı-yapısal ve yapısal olmayan karar verme işlemlerinde destek sağlamak amacıyla, karar modellerine ve verilere olay erişimi sağlayan etkileşimli bir sistemdir (Kroeber 1987, Yozgat 1998).
- Karar Destek Sistemleri, kararın yapısal olmadığı durumlarda karar alma işlemine yardımcı olmak için tasarlanmış, esnek ve etkileşimli bilişim teknolojisi sistemleridir (Hang ve ark. 1998, Yozgat 1998).
- Karar vericinin yerine geçmesinden ziyade onun kararlarını destekleyen, yarı yapısal ve yapısal olmayan problemlerin çözümü için karar vericiye karar vermesinde yardımcı olan etkileşimli sistemleridir (Keen ve Norton 1982, Yozgat 1998).

İki tip KDS vardır: Model-odaklı ve veri-odaklı (Dhar ve Stein 1997). Model-odaklı KDS her şeyden önce, "Şayet…Olursa (What … if)" ve diğer farklı incelemelerin yapılması için bazı modeller kullanan büyük organizasyonel bilgi sistemlerinden bağımsız, tek başına sistemlerdir. Bu tür sistemler genellikle merkezi bilgi sistemi kontrolü altında olmayan son kullanıcı bölümler yada gruplar tarafından geliştirilirler. Bu sistemlerin analiz yetenekleri, modelin kullanımını basitleştirecek iyi bir kullanıcı ara yüzüyle entegrasyonuna bağlıdır. Veri-odaklı KDS, büyük organizasyonel sistemlerde bulunan büyük veri depolarını inceleyen sistemlerdir.

Bu sistemler, daha önceden büyük miktarlardaki verilerde saklı kalan faydalı bilgilerin çıkarılarak, kullanıcılara karar verme desteği sağlayan sistemlerdir. Veri işlem sistemlerinden elde edilen veriler, bu amaç için genellikle veri deposunda toplanırlar (Laudon ve Laudon 2002, Yozgat 1998).

Karar destek sistemleri, yönetim bilişim sistemine ek olarak karar verme işlevinin bilgisayara uygulaması olup insan zekasının, bilgi teknolojisinin ve yazılımın karmaşık problemleri çözmek için etkileşimli olarak bütünleştirilmesidir (Yozgat 1998). Karar destek sistemleri bilgisayar tabanlı yapıyla etkileşimli olarak yöneticilerin karar vermelerine destek sağlarlar. Karar destek sistemleri formal, kapalı, yapılandırılmış, standart sistemler değillerdir. Yöneticilerin ve analizcilerin problemleri anlayıp, değerlendirip çözmelerine yardımcı olacak verilere, modellere, aletlere vb. çeşitli kaynaklara sahiptirler.

Karar Destek Sistemleri'nin dört karakteristik özelliği vardır. Bunlar:

- Sistem özellikle yarı yapılandırılmış veya yapılandırılmamış karar verme görevlerine yardımcı olmak amacıyla dizayn edilmiştir.
- Sistem otomatikleşmiş kararlar dışındaki karar verme problemlerinde de destek sağlar.
- Sistem karar vericinin değişen ihtiyaçlarını da hızla cevaplayabilir.
- Sistem karar vermenin verimliliğini artırırken etkinliğini de artırır (Dock ve Wetherbe 1988, Yozgat 1998).

Belirli modellere uygulanarak çözümlenebilen ve otomatikleştirilebilen kararlar genellikle basit, alışılmış, tekdüze ve yinelemeli kararlardır. Bu tür kararların otomatikleştirilmelerinin sebeplerinden biri de yöneticilerin bu kararlara daha az zaman ayırırken acil ve zorunlu stratejik kararlara daha çok önem verebilmelerini sağlamaktır.

Bu arada, belirli modellere uygulanabilen daha karmaşık modeller de olabilir. Örneğin, sorunlara çözüm önerileri bulunmasında ve çözüm seçeneklerinin karşılaştırılmasından bu modellerden yararlanılabilir. Bu ise yönetici – bilgisayar diyalogunun kurulabilmesi sonucu olmaktadır.

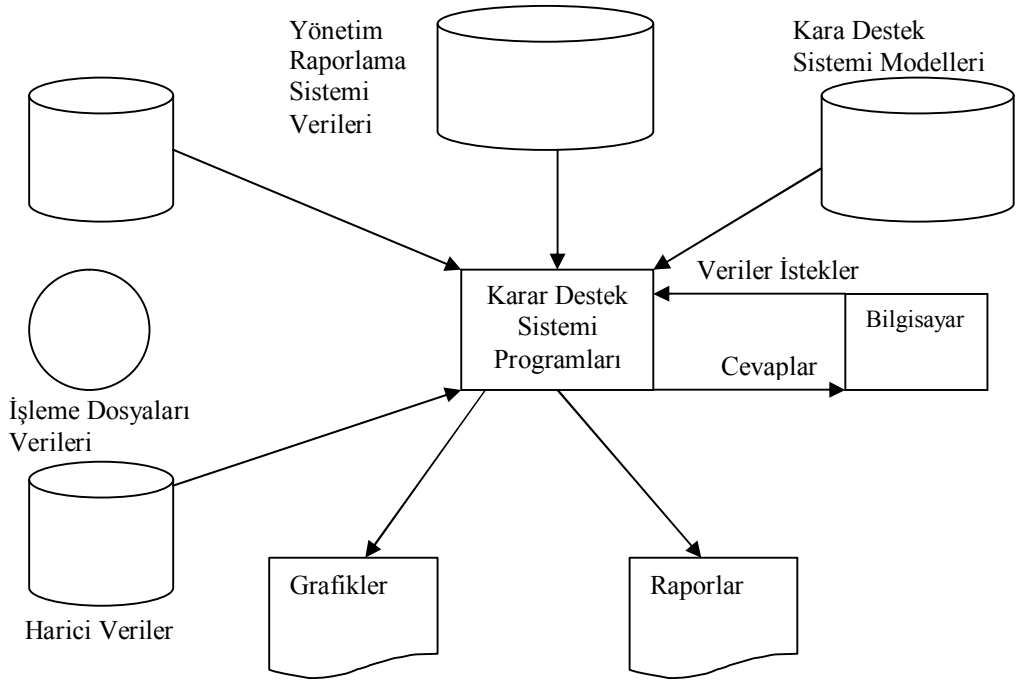
Yönetici – bilgisayar diyalogunun kurulduğu yönetim karar destek sistemlerinde, yöneticiler her sorun için çözüm seçeneklerini formüle eder ve bilgisayara gönderir.

Bilgisayar bu önerileri karşılaştırarak değerlendirir ve sonuçları yöneticiye yollar. Yönetici de değerlendirirken ya öneriler arasında en iyi sonucu veren alternatifi seçer ya da edinilen yeni bilgiler ışığında yeni seçenekler hazırlayarak bilgisayarın değerlendirilmesine sunar. Bu diyalog karar verme için ayrılan süre dolana veya tatmin edici bir sonuç alınana kadar sürdürülebilir (Ülgen 1980, Yozgat 1998).

Karar destek sistemlerinin, geliştirilmiş yönetim bilişim sistemleri olduğu gibi gözden kaçırılmamalıdır. Yönetim bilişim sistemleri geçmişe odaklanmıştır. "Ne gerçekleşti? Ne doğruydu? sorularına yanıt arar. Yönetim bilişim sistemlerinde, çok büyük miktarlardaki veri, planlanan zaman içinde yavaş ve programlı olarak işlenir ve değerlendirilir.

Yönetim bilişim sistemleri karar destek sistemlerinin anahtar parçalarından biridir; ancak kesinlikle onun yerini alamaz. Karar destek sistemleri bugün ve gelecekle ilgilidir. Asıl işlevi yöneticilerin gelecekle ilgili kararlar almalarına yardımcı olup bu kararları organizasyon içinde tartışmalarını sağlayabilmektedir.

Bilgisayar teknolojisi kullanılması sayesinde büyük miktarlardaki veriler hızlı bir şekilde işlenerek karar üzerindeki tüm etkileri belirlenebilir. İşlemlerin hızlı gerçekleşmesi yöneticilerin daha fazla sayıda alternatifi değerlendirebilmesine imkan verir. Şekil 3.5'te karar destek sistemlerinin yapısı gösterilmiştir.



Şekil 3.5. Karar Destek Sistemi Yapısı (Yozgat 1998)

Karar Destek sistemleri uygulamalarının ana işlevleri: (Kroenke 1992, Yozgat 1998)

- Problem dönemi yakından tanıyabilmek,
- Sonuçların, karar değişkenlerindeki değişimleri duyarlılığını tespit etmek,
- Yolları ve trendleri belirlemek,
- Sonuçları tahmin etmek,
- İşletme faaliyetleri için modeller tasarlamak,
- En iyi karışımları hesaplamak,
- Haberleşmeye, kolektif çalışmaya ve ekip çalışmasına olanak sağlamaktır.

Yönetim karar destek sistemlerini kullanan işletmelerde yapılan araştırmalar sonucu sistemin başarısını etkileyen faktörler:

- Kullanıcıların eğitimi,
- Üst yönetimin desteği,
- Uygulamaların yeniliği olarak belirlenmiştir.

Buradaki başarı, karar vermedeki performansın artması ve sistemden duyulan memnuniyet olarak tanımlanmıştır.

Yapılan araştırmalar sonucu işletmelerin destek sistemlerine yaptıkları yatırımın dönüşümüne önem verdiklerini ortaya koymuştur. Bu da işletmelerin karar destek sistemlerini bir anlamda yatırım arası olarak gördüklerini ortaya koymaktadır. Yeterince getiri sağlamadığı düşünüldüğünde gelişme yatırımlarından vazgeçilmektedir (Laudon ve Laudon 1991, Yozgat 1998). Karar destek sistemlerinin kullanımının sadece üst yönetimle sınırlı kalması aksine en alt düzeylere kadar indirgenmiş olması gerekmektedir. Bu sayede daha çok sayıda, türde ve yeni fikirler ortaya çıkararak değerlendirilebilecek, böylelikle de daha etkin kararlar alınabilecektir.

3.4.1. Karar destek sistemlerinin özellikleri

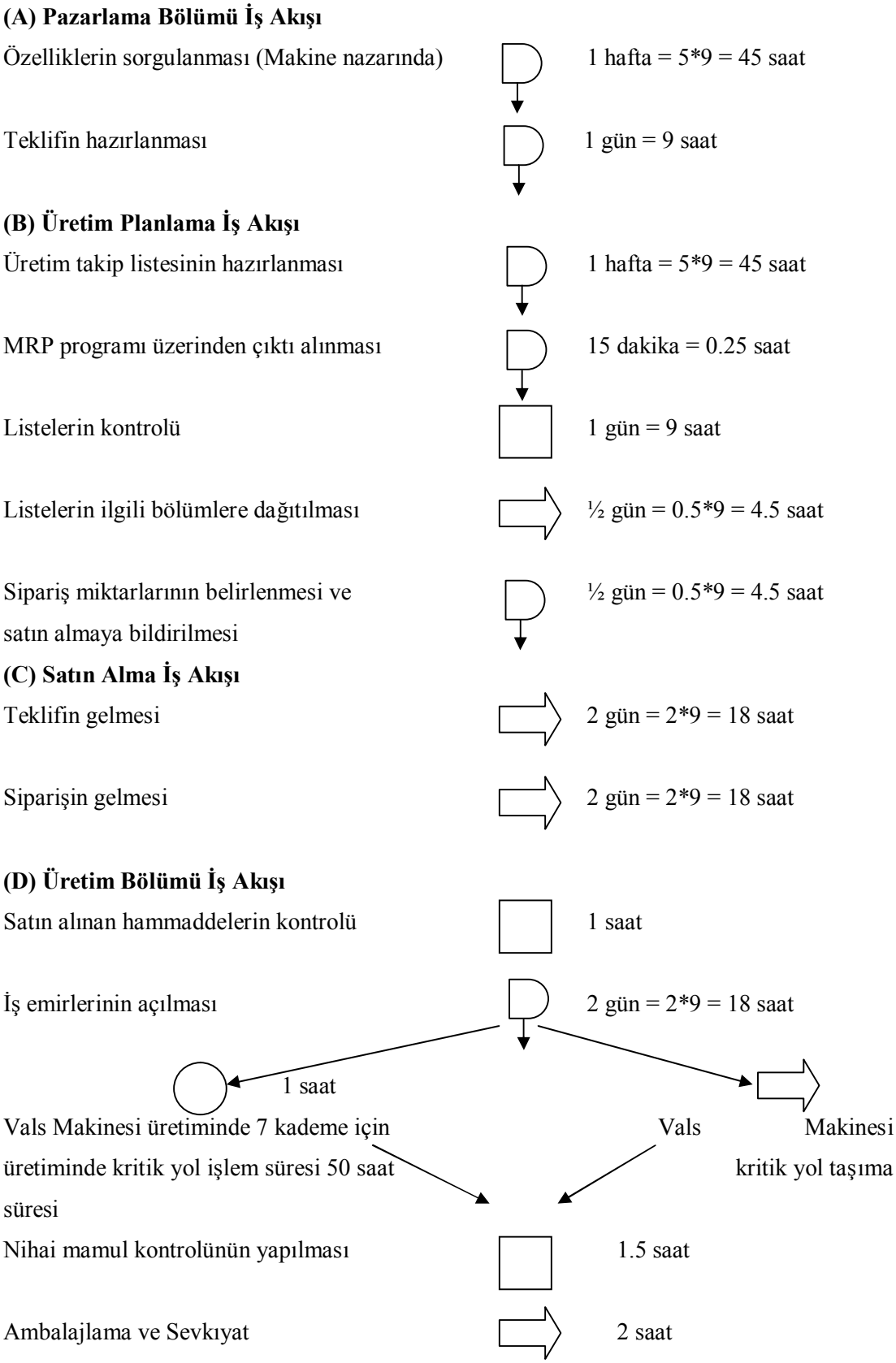
Karar destek sistemi aşağıdaki özelliklere sahiptir (Markas 1999).

- Yarı-yapısal ve yapısal olmayan kararlarda kullanılır.
- Karar vericinin yerine geçmekten ziyade, ona karar vermesinde yardımcı olmaktadır.
- Karar verme prosesinin tüm aşamalarını destekler.
- Kullanıcının kontrolü altındadır.
- Modelden yararlanır.
- Kullanıcı etkileşimlidir.
- Bütün düzeydeki yöneticiler için, gerektiğinde düzeyler arası entegrasyona da destek vererek, karar verme desteği sağlar.
- Birden fazla bağımsız veya birbirine bağımlı kararlar için destek sağlayabilir.

4. ARAŞTIRMA SONUÇLARI

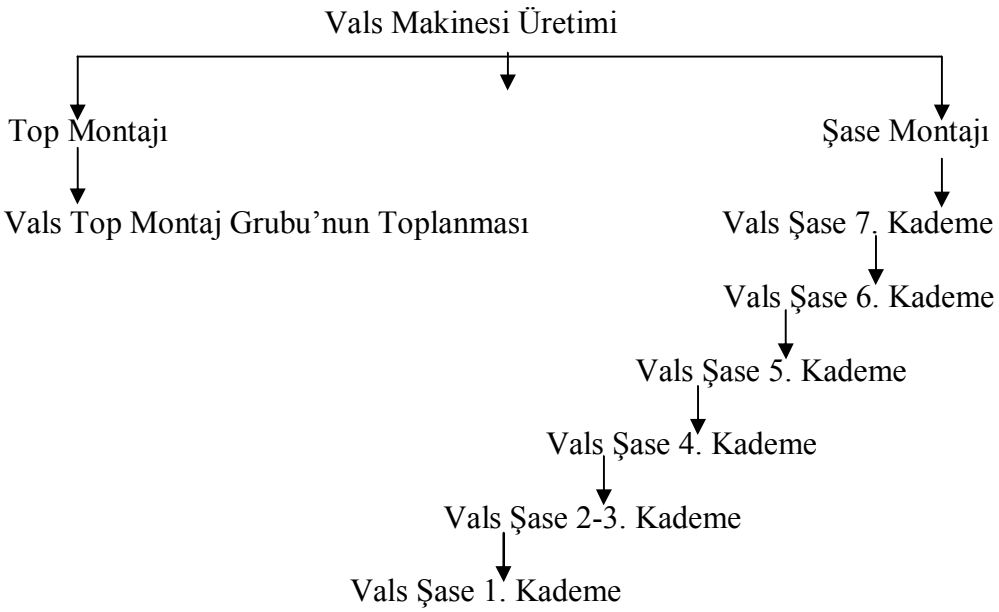
4.1 Vals Makinesi İçin Departmanlar Arası İş Akışı ve Süreleri

İmaş A.Ş.'de üretimi yapılan Vals Makinesi için tam otomatik, yarı otomatik, pnomatik, sensörlü ve sensörsüz olmak üzere makine tip ve modellerine göre iş emirleri açılır. Daha sonra makine alt parçaları iş emirleri açılır. Şase montajı iş emirleri açılır. Şase montajında 7 kademe vardır. 7. Kademe'den sonra nihai ürün kontrolü maksimum 1-1.5 saat arası kadar bir zaman almaktadır. Daha sonra boyahane emirleri açılmakta ve boya sonrası montaja geçilmektedir. Ardından 3-5 saat arası kadar bir süre, mamul kalite kontrol iş emirlerinin hazırlanmasına harcanır. Redüktörün ısınması da 1-2 gün yani 9-18 saat arası kadar bir zaman almaktadır. Ambalajlama ve sevkıyat ise 1'er saat sürer. Şekil 4.1'de Vals Makinesi için departmanlar arası iş akışı ve süreleri gösterilmiştir.



Şekil 4.1. Vals Makinesi İçin Departmanlar Arası İş Akış Süreleri

4.2 Vals Makinesi Üretiminin İncelenmesi



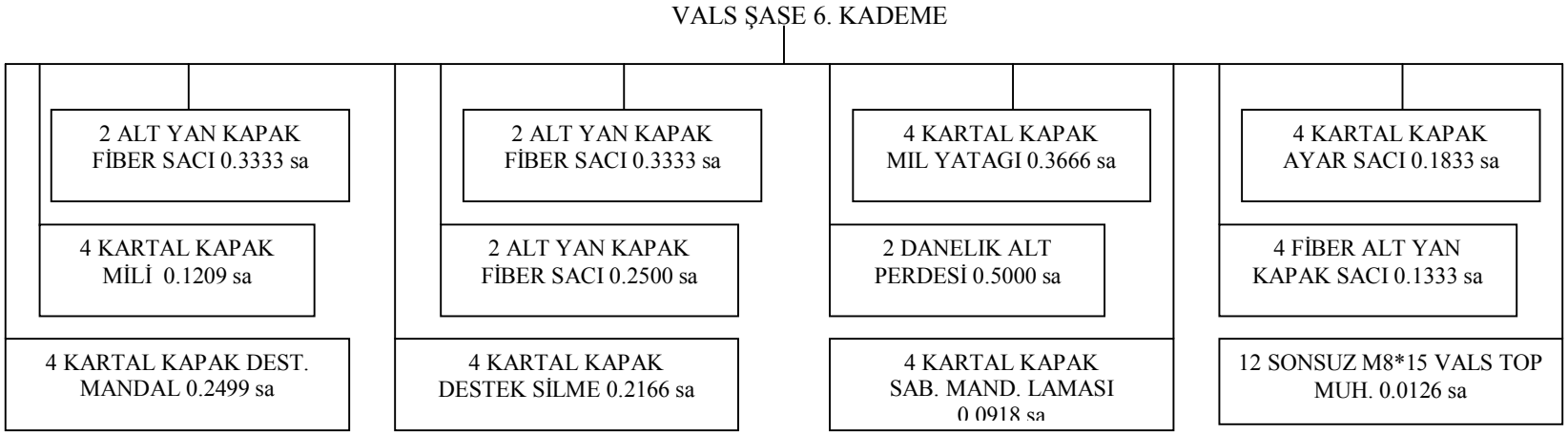
Şekil 4.2. Vals Makinesini Oluşturan Kısımlar

Tablo 4.1. Top Montaj Grubunu Oluşturan Parçaların Adet ve Süre Miktarları

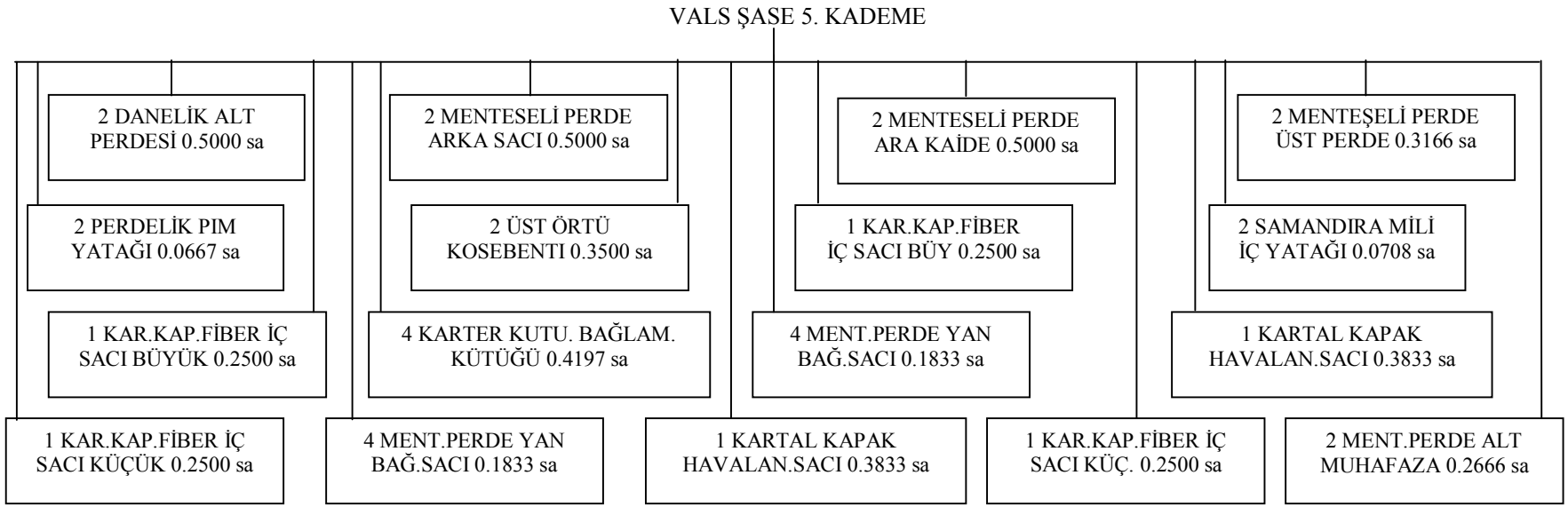
Parça Adı	Adet	Süre (saat)	Parça Adı	Süre (saat)	Adet
Gerdirme Cıvatası	4	2.1075	Silindir Har.Kol Kulaksız	0.4582	2
Yatak Kapağı Uzun	4	0.6122	Sabit Yatak Sağ	2.5054	2
Yatak Kapağı Kısa	4	0.7259	Sabit Yatak Burcu	0.2293	4
Dişli Ara Burcu Orta	4	0.3436	Fırça Alt Tutamağı	0.1125	8
Kasnak Ara Burcu Kısa	2	0.0333	Fırça Şasesi Kırma Montaj	0.1666	2
Flanş Ara Burcu Uzun	2	0.0395	Ayar El Çarkı	0.1742	4
Top Boru Tutturucusu	8	0.0042	Mıka Kapak Aluminyum Prof	0.0750	5
Top Kasnağı	2	1.5671	Mıka Kapak Aluminyum Lama	0.0417	5
Yatak Arka Kapak	8	0.4813	Karter Kutusu	1.6378	2
Gerdirme Kas.Vidalı Perno	2	1.0250	Karter Kutusu Kapağı	1.2730	2
Sabit Pul Yatak Pernosu	4	1.1130	Vida Mıka Kapak Menteşe	0.0161	4
Manivela Kol Burcu	4	0.1312	Hareketli Yat.Sağ Montaj	0.3667	2
Alın Bilyası Burcu	4	1.1417	Hareketli Yat.Sol Montaj	3.3500	2
Top Sıkma Mili	4	1.0542	Sabit Yatak Sol	3.8252	2
Sıkma Konığı	4	2.0667	Fırça Şasesi Liso Montaj	0.1666	2
Sıkma Koniği Gövdesı	4	2.0877	Karter Kutusu Bağl.Cıvata	0.1000	4
Sıkma Koniği Somunu	4	2.1210	Top Boru Tutturucu Pulu	0.2121	8
Top Üst Ayar Mili	4	1.1480	Kayıs Gerdirme Kasnagı	0.3750	2
Tas Yayı	48	1.0404	Hareket.Yat.Man.Kol.Sağ	1.5033	2
Top Ayar Mili Tampon Düz	8	1.0181	Hareket.Yat.Man.Kol.Sol	1.5233	2
Top Ayar Mil Tampon Sivri	8	1.0243	Kızak Gerdir.Catalı	0.8734	4
Silindir Hark.Kol Kulaklı	2	0.5557	Kızak Gerdir.Kütüğü	2.9099	2

Vals Makinesi üretimi top ve şase montajı olmak üzere Şekil 4.2'de gösterildiği gibi iki ana kısımdan meydana gelmektedir. Top Montaj Grubu ise, Tablo 4.1'de isimleri verilen parçalardan oluşmaktadır.

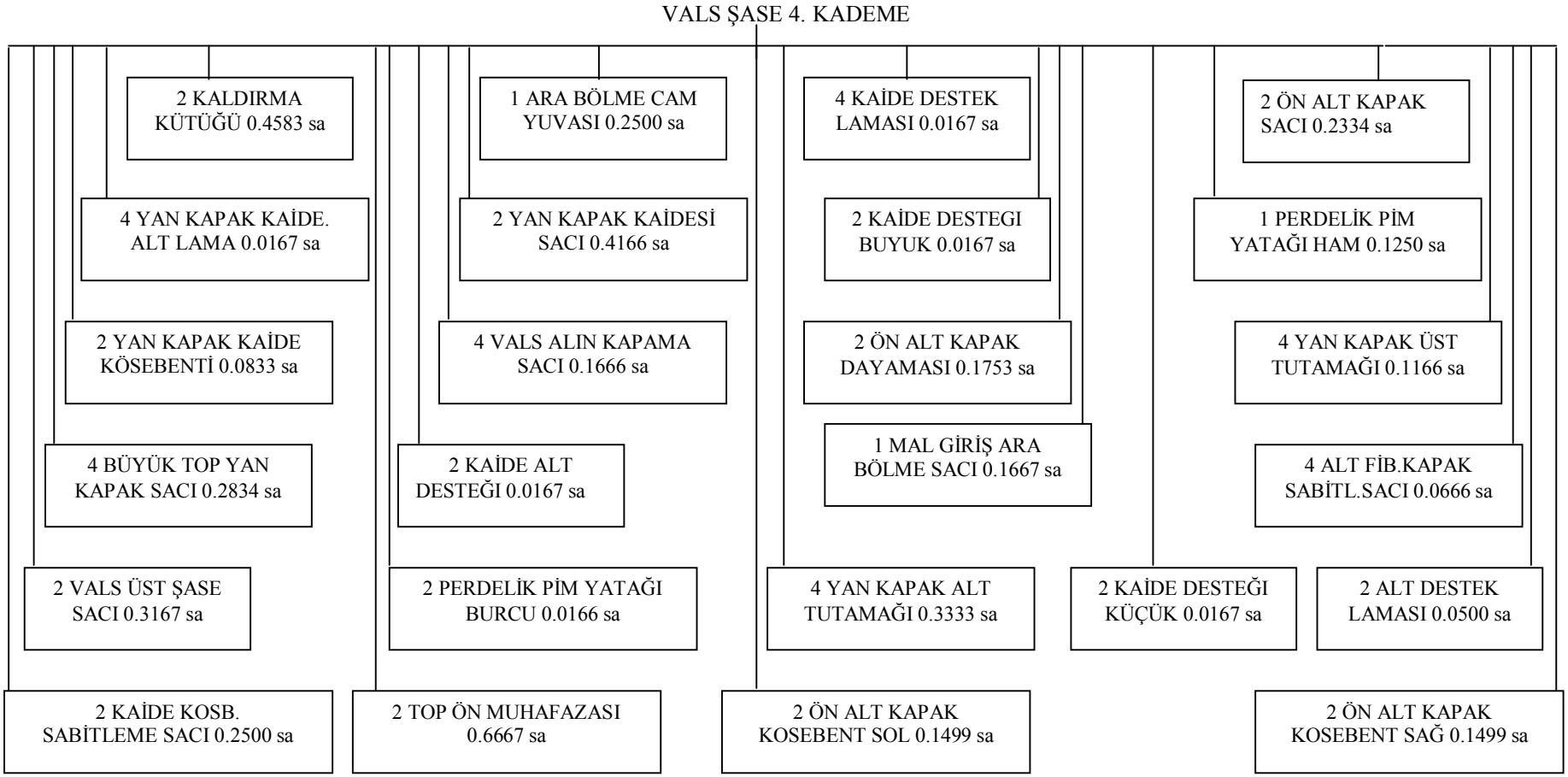
Şase Montajı için 7. Kademe'deki parçaların montajının tamamlanması gerekir. Bunun içinde 6. Kademe'deki parçaların montajı bitirilmiş olmalıdır. Aynı şekilde 6. Kademe için 5. Kademe'nin, 5. Kademe için 4. Kademe'nin, 4. Kademe için 2-3. Kademe'nin, 2-3. Kademe için de 1. Kademe'nin montaj parçaları birleştirilmelidir.



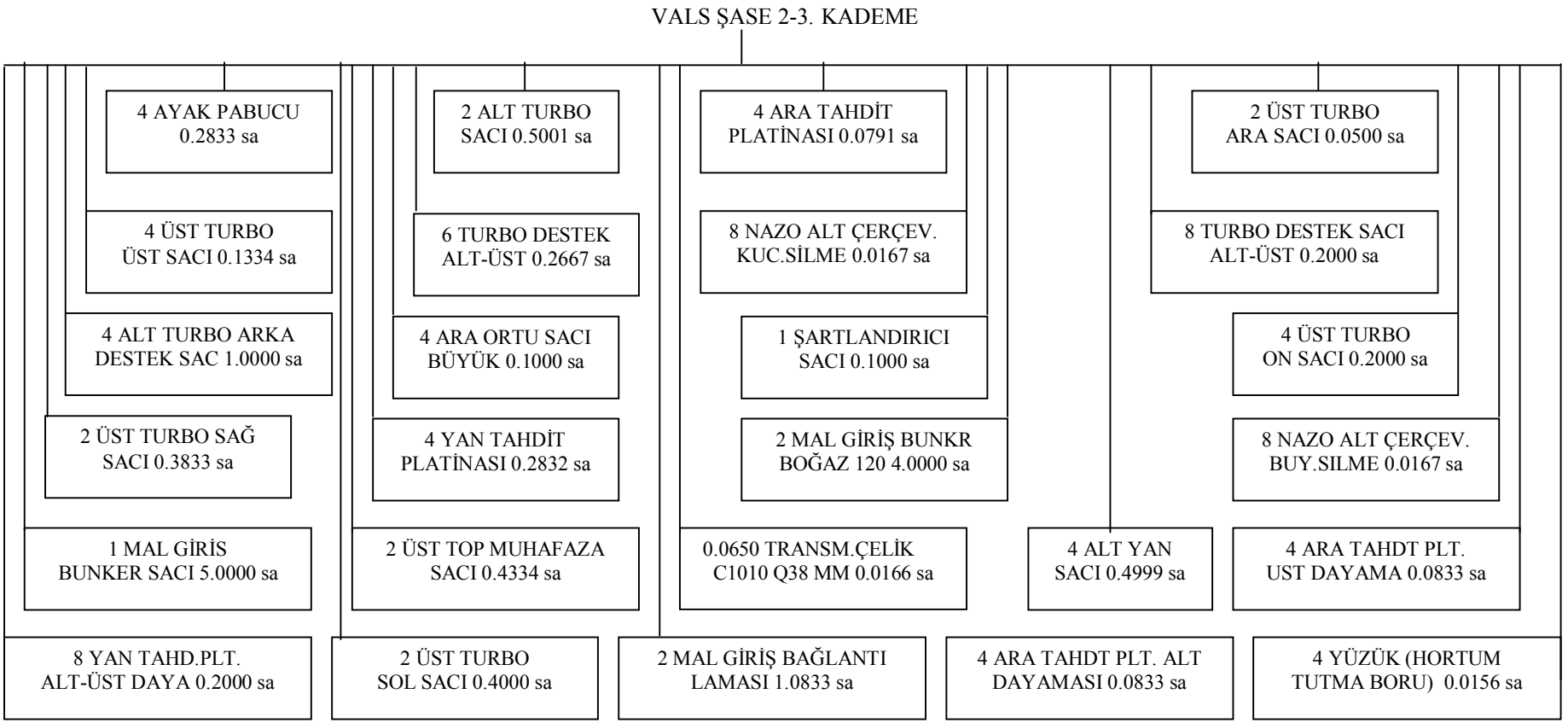
Şekil 4.3. Vals Şase 6. Kademe'de Montajı Yapılan Parçalar



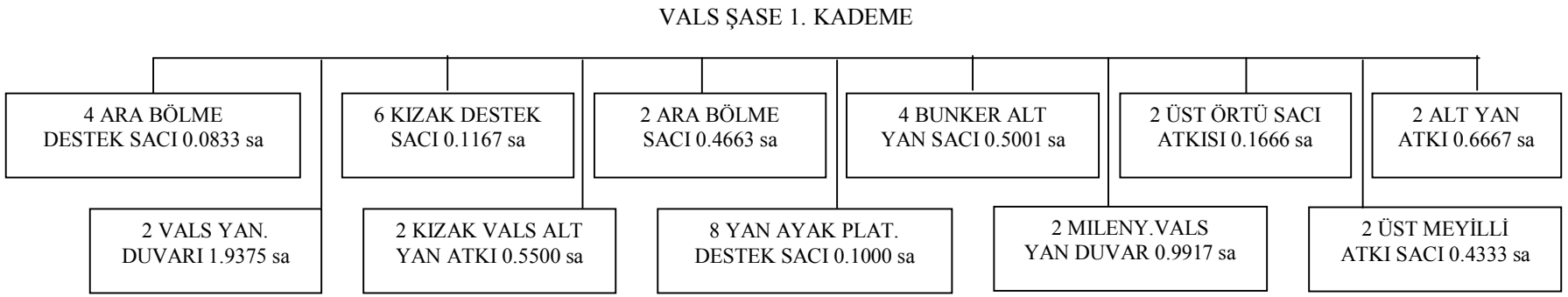
Şekil 4.4. Vals Şase 5. Kademe'de Montajı Yapılan Parçalar



Şekil 4.5. Vals Şase 4. Kademe'de Montajı Yapılan Parçalar



Şekil 4.6. Vals Şase 2-3. Kademe'de Montajı Yapılan Parçalar



Şekil 4.7. Vals Şase 1. Kademe'de Montajı Yapılan Parçalar

Şekil 4.3-4.7 arasındaki kutucuklarda yer alan montaj parçalarının sol tarafındaki değer montaj adedini, sağ tarafındaki değer ise saat cinsinden montaj süresini göstermektedir. 7. Kademe'de sadece 6. Kademe'de montajı tamamlanan parçaların ardından Vals Şasesi kaynak edilmekte ve bu işlem 0.6500 saatlik bir zaman almaktadır.

Bu tezin araştırma sonuçları bölümünde Vals makinesi üretiminde Vals şase 1. kademede montajı yapılan parçalardan ara bölme sacının talebi (net ihtiyacı) ele alınmıştır. Ara bölme sacının sipariş miktarını belirlemek ve maliyeti hesaplamak için kullanılan tekniklerde hazırlık maliyeti 1500 para birimi, stok taşıma maliyeti ise 49,2 birim/yıl olarak alınmıştır.Bu malzemenin aylık talebine ilişkin firmadan alınan veriler aşağıdaki gibidir.

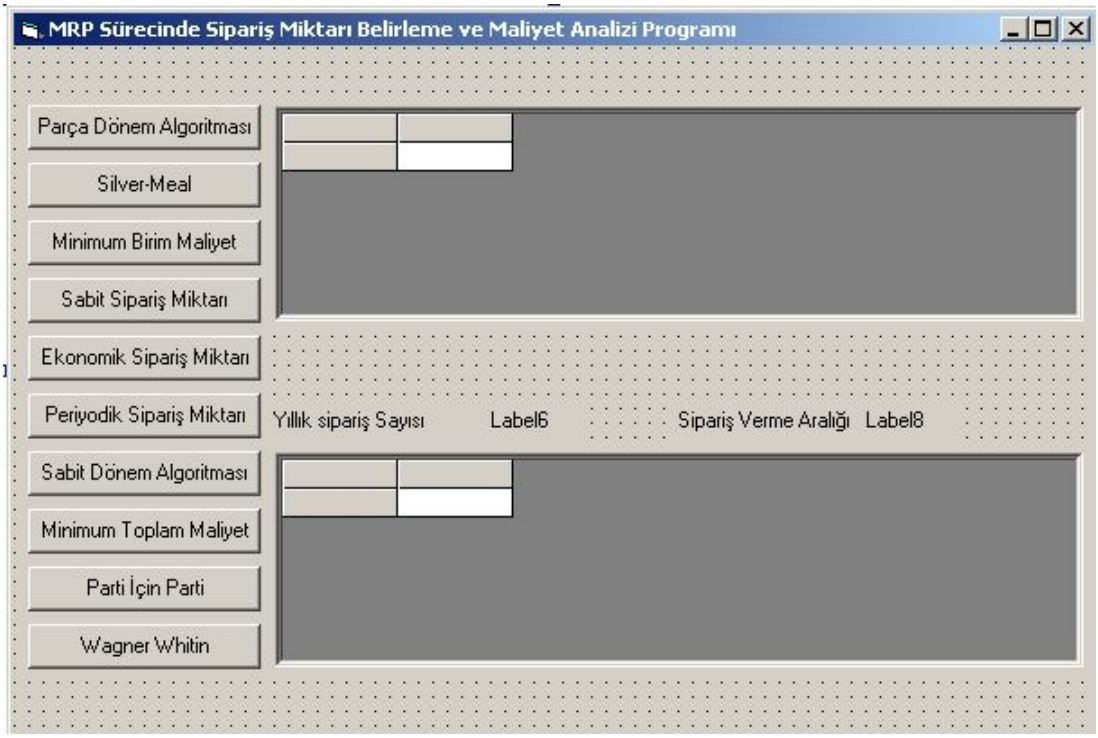
Tablo 4.2. Ara Bölme Sacının Aylık Talepleri

Dönem	1	2	3	4	5	6	7	8	9	10	11	12	Toplam
Net İhtiyaçlar	775	475	725	650	575	725	775	800	825	850	825	750	8750

4.3. Vals Makinesi İçin Sipariş Miktarı ve Toplam Maliyet Hesabı

4.3.1. Kullanılan yazılımın arayüzü

Tablo 4.3 Sipariş Miktarı Belirleme ve Maliyet Analizi Programı Arayüzü



4.3.2. Sabit sipariş miktarı yöntemi ile çözüm

Yazılım programı işletilip "sabit sipariş miktarı yöntemi" ile aylık net ihtiyaçlar girildiğinde bulunan sonuçlar (her ay için verilen sipariş miktarı, stok taşıma maliyeti, hazırlık maliyeti ve toplam maliyet) Tablo 4.4'te gösterilmiştir.

Tablo 4.4 Sabit Sipariş Miktarı Yöntemi İle Sonuçlar

Aylar	1	2	3	4	5	6	7	8	9	10	11	12	Toplam
Net İhtiyaç	775	475	725	650	575	725	775	800	825	850	825	750	8750
Verilen Sipariş	1500		1500		1500		1500		1500		1500		9000
Stok Taşıma Maliyeti	0	725	250	1025	375	1300	575	1300	500	1175	325	1000	8550
Hazırlık Maliyeti	1500		1500		1500		1500		1500		1500		9000
Toplam Maliyet													17550

4.3.3. Net ihtiyaç kadar sipariş yöntemi ile çözüm

Yazılım programı işletilip "net ihtiyaç kadar sipariş yöntemi" ile aylık net ihtiyaçlar girildiğinde bulunan sonuçlar (her ay için verilen sipariş miktarı, stok taşıma maliyeti, hazırlık maliyeti ve toplam maliyet) Tablo 4.5'te gösterilmiştir.

Tablo 4.5 Net İhtiyaç Kadar Sipariş Yöntemi İle Sonuçlar

Aylar	0	1	2	3	4	5	6	7	8	9	10	11	12	Toplam
Net İhtiyaç		775	475	725	650	575	725	775	800	825	850	825	750	8750
Verilen Sipariş	775	475	725	650	575	725	775	800	825	850	825	750		8750
Stok Taşıma Maliyeti	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Hazırlık Maliyeti	1500	1500	1500	1500	1500	1500	1500	1500	1500	1500	1500	1500	1500	18000
Toplam Maliyet														18000

4.3.4. Ekonomik sipariş miktarı yöntemi ile çözüm

Hazırlık maliyeti (S): 1500 para birimi

Stok taşıma maliyeti (I): 49,2 para birimi (yıllık)

Yıllık kullanım miktarı (U): 8750 adet

$EOQ = \sqrt{(2*U*S)/(I)} = \sqrt{(2*8750*1500)/(49,2)} = 730$ adet (3.4)

Eşitlik 3.4'te yöntemin hesaplanışına ait formül verilmiştir. Yazılım programı işletilip "ekonomik sipariş miktarı yöntemi" ile aylık net ihtiyaçlar girildiğinde bulunan sonuçlar (her ay için verilen sipariş miktarı, stok taşıma maliyeti, hazırlık maliyeti ve toplam maliyet) Tablo 4.6'da gösterilmiştir.

Tablo 4.6 Ekonomik Sipariş Miktarı Yöntemi İle Program Çıktısı

Aylar	1	2	3	4	5	6	7	8	9	10	11	12	Toplam
Net İhtiyaç	775	475	725	650	575	725	775	800	825	850	825	750	8750
Verilen Sipariş	730		730		730		730		730		730		4380
Stok Taşıma Maliyeti		725	250	1025	375	1300	575	1300	500	1175	325	1000	8550
Hazırlık Maliyeti	1500		1500		1500		1500		1500		1500		9000
Toplam Maliyet													17550

4.3.5. Sabit dönem algoritması ile çözüm

Yazılım programı işletilip "sabit dönem algoritması" ile aylık net ihtiyaçlar girildiğinde bulunan sonuçlar (her ay için verilen sipariş miktarı, stok taşıma maliyeti, hazırlık maliyeti ve toplam maliyet) Tablo 4.7'de gösterilmiştir.

Tablo 4.7 Sabit Dönem Algoritması Yöntemi İle Sonuçlar

Aylar	1	2	3	4	5	6	7	8	9	10	11	12	Toplam
Net İhtiyaç	775	475	725	650	575	725	775	800	825	850	825	750	8750
Verilen Sipariş	1975			1950			2400			2425			8750
Stok Taşıma Maliyeti	0	1200	725	0	1300	725	0	1625	825	0	1575	750	8725
Hazırlık Maliyeti	1500			1500			1500			1500			6000
Toplam Maliyet													14725

4.3.6. Periyodik sipariş miktarı yöntemi ile çözüm

Ekonomik sipariş miktarı örneğinde EOQ = 730 olarak bulunmuştu.

Bir yıllık dönem sayısı = 12

Yıllık talep = 8750

8750/730 ≈ 6 (Yıllık sipariş sayısı)

12/6 = 2 (Sipariş verme aralığı)

Yazılım programı işletilip "periyodik sipariş miktarı yöntemi" ile aylık net ihtiyaçlar girildiğinde bulunan sonuçlar (her ay için verilen sipariş miktarı, stok taşıma maliyeti, hazırlık maliyeti ve toplam maliyet) Tablo 4.8'de gösterilmiştir.

Tablo 4.8 Periyodik Sipariş Miktarı Yöntemi İle Sonuçlar

Aylar	1	2	3	4	5	6	7	8	9	10	11	12	Toplam
Net İhtiyaç	775	475	725	650	575	725	775	800	825	850	825	750	8750
Verilen Sipariş	1250		1375		1300		1575		1675		1575		8750
Stok Taşıma Maliyeti	0	475	0	650	0	725	0	800	0	850	0	750	4250
Hazırlık Maliyeti	1500		1500		1500		1500		1500		1500		9000
Toplam Maliyet													13250

Yıllık sipariş Sayısı 6 Sipariş Verme Aralığı 2

4.3.7. En düşük birim maliyet yöntemi ile çözüm

Yazılım programı işletilip "en düşük birim maliyet yöntemi" ile aylık net ihtiyaçlar girildiğinde Tablo 4.9'da bulunan sonuçlar (stokta taşınan dönem sayısı, muhtemel sipariş miktarı, birim hazırlık maliyeti) Tablo 4.10'da ise sipariş tablosu (her ay için verilen sipariş miktarı, stok taşıma maliyeti, hazırlık maliyeti ve toplam maliyet) gösterilmiştir.

Tablo 4.9 En Düşük Birim Maliyet Yöntemi İle Sonuçlar

Dönem	Net İhtiyaç	Stokta Taş. Dön.	Muht. Sip. Mikt.(1)	Taşıma Parti İçin(2)	Mal. Bir. İçin(2/1)	Bir. Haz. Mal.(Hm/1)	Bir. Mal.
1	775	0	775	0	0	1,94	1
2	475	1	1250	475	0,38	1,2	1
3	725	2	1975	1450	0,73	0,76	0
1	725	0	725	0	0	2,07	2
2	650	1	1375	650	0,47	1,09	1
3	575	2	1950	1150	0,59	0,77	0
1	575	0	575	0	0	2,61	2
2	725	1	1300	725	0,56	1,15	1
3	775	2	2075	1550	0,75	0,72	0
4	800	3	2875		0	0,52	0
1	800	0	800	2400	3	1,88	4
2	825	1	1625	825	0,51	0,92	0
3	850	2	2475	1700	0,69	0,61	0
1	850	0	850	0	0	1,76	1
2	825	1	1675	825	0,49	0,9	0
3	750	2	2425	1500	0,62	0,62	0
1	750	0	750	0	0	2	2

Tablo 4.10 En Düşük Birim Maliyet Sipariş Tablosu

Aylar	1	2	3	4	5	6	7	8	9	10	11	12	Toplam
Net İhtiyaç	775	475	725	650	575	725	775	800	825	850	825	750	8750
Verilen Sipariş	1250		1375		2075			1625		1675		750	8750
Stok Taşıma Maliyeti		475	0	650	0	1500	775	0	825	0	825	0	5050
Hazırlık Maliyeti	1500		1500		1500			1500		1500		1500	9000
Toplam Maliyet													14050

4.3.8. En düşük toplam maliyet yöntemi ile çözüm

EPP = En düşük toplam maliyet

S = Hazırlık maliyeti = 1500 para birimi

Ip = Dönemsel stok taşıma maliyeti = 49,2/12 = 4,1 birim

C = Birim maliyet = 1 para birimi

EPP = S/(Ip*C) = 1500/(4,1 *1) ≈ 366

Tablo 4.11'deki "muhtemel parti büyüklüğü" sütunundaki değerlerden bulunan EPP değeri olan 366'ya en yakın olanları yani dönem 1'lere ait muhtemel parti büyüklükleri sipariş miktarları olarak seçilmiştir.

Tablo 4.12'de ise "en düşük toplam maliyet yöntemi" için sipariş tablosu (her ay için verilen sipariş miktarı, stok taşıma maliyeti, hazırlık maliyeti ve toplam maliyet) gösterilmiştir.

Tablo 4.11 En Düşük Toplam Maliyet Yöntemi İle Sonuçlar

Dönem	Net İhtiyaç	Stokta Taşındığı Dönem Sayısı	Muhtemel Parti Büyüklüğü	Parça-Dönem
1	775	0	775	0
2	475	1	1250	475
3	725	2	1975	1925
4	650	3	2625	3875
1	650	0	650	0
2	575	1	1225	575
3	725	2	1950	1450
1	725	0	725	0
2	775	1	1500	775
3	800	2	2300	1600
1	800	0	800	0
2	825	1	1625	825
3	850	2	2475	1700
1	850	0	850	0
2	825	1	1675	825
3	750	2	2425	2325
1	750	0	750	0

Tablo 4.12 En Düşük Toplam Maliyet Sipariş Tablosu

Aylar	1	2	3	4	5	6	7	8	9	10	11	12	Toplam
Net İhtiyaç	775	475	725	650	575	725	775	800	825	850	825	750	8750
Verilen Sipariş	1975			1225		1500		1625		1675		750	8750
Stok Taşıma Maliyeti		1200	725	0	575	0	775	0	825	0	825	0	4925
Hazırlık Maliyeti	1500			1500		1500		1500		1500		1500	9000
Toplam Maliyet													13925

4.3.9. Parça dönem algoritması

Tablo 4.13'te "parça dönem algoritması" ile bazı aylarda verilmesi gereken sipariş miktarları, Tablo 4.14'de ise bu algoritma ile yapılan maliyet hesabı (her ay için verilen sipariş miktarı, stok taşıma maliyeti, hazırlık maliyeti ve toplam maliyet) gösterilmiştir.

Tablo 4.13 Parça Dönem Algoritması İle Sipariş Miktarı

Aylar	1	2	3	4	5	6	7	8	9	10	11	12	Toplam
Net İhtiyaç	775	475	725	650	575	725	775	800	825	850	825	750	8750
Parça-Dönem	0	1	2	3	1	2	1	2	1	2	1	2	
		475	725	650	575	725	775	800	825	850	825	750	
Parça-Dönem(kümülatif)	0	475	1925	3875	575	2025	575	2175	575	1700	575	1500	
Sipariş Miktarı	1975			1225		1500		1625		1675		750	8750

Tablo 4.14 Parça Dönem Algoritması İle Maliyet Hesabı

Aylar	1	2	3	4	5	6	7	8	9	10	11	12	Toplam
Net İhtiyaç	775	475	725	650	575	725	775	800	825	850	825	750	8750
Verilen Sipariş	1975			1225		1500		1625		1675		750	8750
Stok Taşıma Maliyeti	0	1200	725	0	575	0	775	0	825	0	825	0	4925
Hazırlık Maliyeti	1500			1500		1500		1500		1500		1500	9000
Toplam Maliyet													13925

4.3.10. Silver-Meal sezgisel algoritması ile çözüm

Yazılım programı işletilip "Silver-Meal Algoritması" ile aylık net ihtiyaçlar girildiğinde Tablo 4.15'te her ay için verilen sipariş miktarı, stok taşıma maliyeti, hazırlık maliyeti ve toplam maliyet gösterilmiştir.

Tablo 4.15 Silver-Meal Yöntemi İle Sipariş Miktarı ve Maliyet Hesabı

Aylar	1	2	3	4	5	6	7	8	9	10	11	12	Toplam
Net İhtiyaç	775	475	725	650	575	725	775	800	825	850	825	750	8750
Verilen Sipariş	1250		725	1225		725	775	800	825	850	825	750	8750
Stok Taşıma Maliyeti	0	475	0	1500	575	0	0	0	0	0	0	0	2550
Hazırlık Maliyeti	0		1500	1500		1500	1500	1500	1500	1500	1500	1500	13500
Toplam Maliyet													16050

4.3.11. Wagner-Whitin algoritması ile çözüm

Yazılım programı işletilip "Wagner-Whitin Algoritması" ile aylık net ihtiyaçlar girildiğinde Tablo 4.16'da bulunan sonuçlar (her ay için verilen sipariş miktarı, stok taşıma maliyeti, hazırlık maliyeti ve toplam maliyet), Tablo 4.17'de ise siparişlerin tespiti ve stok maliyetleri gösterilmiştir.

Tablo 4.16 Wagner-Whitin Algoritması İle Sonuçlar

Aylar	1	2	3	4	5	6	7	8	9	10	11	12	Toplam
Net İhtiyaç	775	475	725	650	575	725	775	800	825	850	825	750	8750
Verilen Sipariş	1975			1950			4825						8750
Stok Taşıma Maliyeti		1200	725	0	1300	725	0	4050	3250	2425	1575	750	16000
Hazırlık Maliyeti	0			0			0						0
Toplam Maliyet													16000

Tablo 4.17 Wagner-Whitin Algoritması İle Siparişlerin Belirlenmesi ve Stok Maliyetleri

Aylar	1	2	3	4	5	6	7	8	9	10	11	12
Talep	775	475	725	650	575	725	775	800	825	850	825	750
Hiz. Sipariş Maliyeti	1500	1500	1500	1500	1500	1500	1500	1500	1500	1500	1500	1500
Stok Taşıma Maliyeti	1	1	1	1	1	1	1	1	1	1	1	1
Stokta Taşınan Dönem Sayısı	0	1	2	3	4	5	6	7	8	9	10	11
	1500	475	1450	1950	2300	3625	4650	5600	6600	7650	8250	8250
		1500	725	1300	1725	2900	3875	4800	5775	6800	7425	7500
			1500	650	1150	2175	3100	4000	4950	5950	6600	6750
				1500	575	1450	2325	3200	4125	5100	5775	6000
					1500	725	1550	2400	3300	4250	4950	5250
						1500	775	1600	2475	3400	4125	4500
							1500	800	1650	2550	3300	3750
								1500	825	1700	2475	3000
									1500	850	1650	2250
										1500	825	1500
											1500	750
												1500
Aylar	1	2	3	4	5	6	7	8	9	10	11	12
	1500	1975	3425	5375	7675	11300	15950	21550	28150	35800	44050	52300
		1500	2225	3525	5250	8150	12025	16825	22600	29400	36825	44325
			1500	2150	3300	5475	8575	12575	17525	23475	30075	36825
				1500	2075	3525	5850	9050	13175	18275	24050	30050
					1500	2225	3775	6175	9475	13725	18675	23925
						1500	2275	3875	6350	9750	13875	18375
							1500	2300	3950	6500	9800	13550
								1500	2325	4025	6500	9500
									1500	2350	4000	6250
										1500	2325	3825
Aylar	1	2	3	4	5	6	7	8	9	10	11	12
	1500	1975	3425	5375	7675	11300	15950	21550	28150	35800	44050	52300
		3000	3725	5025	6750	9650	13525	18325	24100	30900	38325	45825
			3475	4125	5275	7450	10550	14550	19500	25450	32050	38800
				4925	5500	6950	9275	12475	16600	21700	27475	33475
					6875	7600	9150	11550	14850	19100	24050	29300
						9175	9950	11550	14025	17425	21550	26050
							12800	13600	15250	17800	21100	24850
								17450	18275	19975	22450	25450
									23050	23900	25550	27800
										29650	30475	31975
											37300	38050
												45550

4.4. MRP Sipariş Hesaplamalarının Karşılaştırılması

Tablo 4.18'de her parti-hacimlendirme yöntemi için yukarıda hesaplanan toplam maliyet değerlerine yer verilmiş bunlardan en düşük toplam maliyeti veren "periyodik sipariş miktarı yöntemi" altı çizilerek vurgulanmıştır.

Tablo 4.18. MRP Sipariş Hesaplamalarının Karşılaştırılması

Sipariş Miktarı Hesaplama Algoritmaları	Sonuçlar
Sabit sipariş miktarı	17550
Net ihtiyaç kadar sipariş	18000
Ekonomik sipariş miktarı	17550
Sabit dönem algoritması	14725
Periyodik sipariş miktarı	**13250**
En düşük birim maliyet	14050
En düşük toplam maliyet	13925
Parça dönem algoritması	13925
Silver-Meal sezgisel	16050
Wagner-Whitin algoritması	16000

Burada, sonuç olarak "periyodik sipariş miktarı yöntemi"nin bulunması itibariyle siparişler arasında bir ekonomik zaman aralığı hesaplanarak, ortalama talep oranı tarafından bölünen ekonomik sipariş miktarı olarak tanımlanan parti hacimleri bulunmuştur. Bu yöntemde ilk dönemden başlanarak kendisi dahil bir sonraki dönemin siparişi birlikte verilmektedir. Yalnız bir dönem sipariş verildiğinde bunun hemen ardındaki dönem sipariş verilmemektedir.

5. SONUÇ

MRP'nin başarı ile uygulanmasında iki faktör önem taşımaktadır. Birincisi tedarik kaynaklarının güvenilir ve dakik çalışmasıdır. Gecikme payları çok küçük olduğundan tedarikte en küçük aksaklıklar tüm üretim sisteminin durmasına sebep olabilir. İkinci faktör, malzeme ihtiyaç planlaması sistemi için gerekli olan büyük bilgi işlem kapasitesidir. Bunun için MRP programı mutlaka bilgisayar desteği ile uygulanmalıdır. Bu sebepten dolayı tezin uygulama kısmında da tamamen yeni bir program yazılmış ve sipariş miktarı belirleme ve maliyet hesabı bu yazılım üzerinden yapılarak rapor edilmiştir.

Parti hacimleri çok küçük alınırsa sıklıkla hazırlıklara ihtiyaç duyulacak ve makineler yüksek oranda kullanılacaktır. Bu da uzun bekleme sürelerine sebep olacaktır. Ancak parti hacimleri çok büyük alınırsa da makineler aynı parçayı daha uzun sürede işleyecektir. Bu, parça miktarlarının yönetiminde problemlere ve de genelde yüksek envanterlere neden olur. Performansın en iyi duruma gelmesi için sipariş bırakma zamanının bileşenlerin tamamlanma zamanları ile tutarlı olması gerektiği unutulmamalıdır. Bir Malzeme İhtiyacı Planlaması sürecindeki parti-hacimlendirme probleminde uygulandığı zaman en az envanter stokuna ve en iyi gecikme zamanına sahip sezgisel metot seçilmelidir.

Çalışmada geçen ekonomik parti çizelgeleme probleminde sezgisel yöntemin aşırı envantere neden olmamak için bazen üretim tesisini aylak tutması önemle üzerinde durulması gereken bir konudur.

Tezde bahsedilen parti-hacimlendirme yöntemleri analiz edildiğinde; yüksek talep seviyelerinde kapasitelendirilmemiş tekniklerin (Parti-için-parti, Sabit periyot algoritması, Minimum birim maliyet ve Silver-Meal) toplam envanter kıyaslanabilir ölçüde dağıttığı görülmektedir. Talep düşükse bütün parti-hacimlendirme yöntemlerinin toplam envanter maliyetleri Parti-için-parti yöntemininkine yaklaşmaktadır. Bu nedenle Parti-için-parti yöntemi matematiksel sadeliği açısından yüksek talep seviyelerinde genelde tercih edilir.

İmaş A.Ş.'de uygulaması yapılan çalışmada 10 parti-hacimlendirme yöntemi Vals makinesinin talepleri dikkate alınarak verilecek aylık sipariş miktarları bulunmuş ve toplam maliyetler hesaplanmıştır. Tablo 4,18'de bu sonuçlar kıyaslanmış ve tablodan da görüldüğü gibi "periyodik sipariş miktarı yöntemi" en uygun yöntem olarak seçilmiş ve İmaş A.Ş.'ye vals makinesi tedarik sürecinde sipariş miktarı belirlerken bu yöntemin uygulanması önerilmiştir.

Malzeme ihtiyaç planlaması sistemi de, araç olarak kullanılan bilgisayar yazılımı da kendisine sunulan verilerden hareketle sonuca ulaşır. İşte bu sebeple verileri doğruluğu ve yeterliliği sistem için en önemli etken haline gelmektedir. Bu nedenle de yönetimin sistem hakkında bilgilendirilmesi ve sisteme gereken desteği sağlamalıdır. Gelecek araştırmalara kalan ise parti hacimlerinin ne zaman sık hesaplanacağı ve mevcut başlangıç stoku ile nasıl uyarlanacağıdır.

6. KAYNAKLAR

Acar, N. 1991. MRP sistemi. MPM Yayınları, No: 323, 2. Baskı, Ankara

Acar, N. 1999. Malzeme İhtiyaç Planlaması. MPM, 5. Baskı, Ankara

Baker, K. R. 1989. Lot-sizing procedures and a standard data set: a reconciliation of the literature. Journal of Manufacturing and Operations Management, 2: 199–221

Berry, W. L. 1972. Lot sizing procedures for requirements planning systems: a framework for analysis. Production and Inventory Management, 13:19–34

Çelikçapa, F., Sarsılmaz, M. 1999. ERP – İşletme kaynaklarının dünü, bugünü ve yarını. II. Ulusal Endüstri-İşletme Mühendisliği Kurultayı

Çelikçapa, F. O. 2000. Üretim Yönetimi ve Teknikleri. Alfa Basım Yayım Dağıtım

Çetinkaya, T. 1988. Malzeme İhtiyaç Planlaması. Seri Üretimde Üretim Planlama Semineri, TÜSSİDE, Kocaeli

Dağlı, C. 1987. Malzeme İhtiyaç Planlaması Sistemi Tasarımı, Ankara

DeMatteis, J. J. 1968. An economic lot-sizing technique: the part-period algorithm. IBM Systems Journal 7:30–8.

Dock, V. T., Wetherbe, J. C. 1988. Computer Information Systems for Business. West Pub. Co., St. Paul

Gorham, T. 1968. Dynamic order quantities. Production and Inventory Management 9:75–81.

Güneş, M., Firuzen, A. R., Firuzen, E. 1999. Tam Zamanında Üretim (JIT) Ortamında Stok Kontrolü ve Toplam Kalite Yönetimi. Yöneylem Araştırması Yardımcı Ders Kitabı, Barış Yayınları, Fakülteler Kitabevi

Haag, S., Cummings, M., Dawkins, J. 1998. Management Information Systems for the Information Age, Irwin/McGraw Hill Publishing Co

Haddock, J, Hubicki, D. E. 1989. Which lot - sizing techniques are used in material requirements planning?. Production and Inventory Management, 30:53–6

Hamarat, Ş. 1986. Malzeme İhtiyaç Planlaması. Yüksek Lisans Tezi Gazi Üniversitesi Sosyal Bilimler Enstitüsü, Ankara

Ho, J. C., Solis, A. O., Chang, Y. L. 2007. An evaluation of lot-sizing heuristics for deteriorating inventory in material requirements planning systems, Computers & Operations Research 34, 2562 – 2575.

Hu, J., Munson C. L. 2002. Dynamic demand lot - sizing rules for incremental quantity discounts. Journal of the Operational Research Society, 53: 855–63

İlgenli, E. 1989. Tümosan Türk Sanayi ve Ticaret A.Ş.'de Üretim Planlaması ve Kontrolü, Yüksek Lisans Tezi, Konya

Keen, P. G. W., MORTON, M. S. 1982. Decision Support Systems: An Organizational Perspective. Reading, MA: Addison-Wesley

KOBU, B. 1999. Üretim Yönetimi. İstanbul Üniversitesi İşletme Fakültesi, İşletme İktisadi Enstitüsü Araştırma ve Yardım Vakfı, yayın no: 04, Avcıol Basım-Yayın

Kroeber, D. W., WATSON, H. J. 1987. Computer Based Information Systems: A Management Approach

Kroenke, D., Hatch R. 1992. Business Information Systems. 5. Baskı, McGraw-Hill, New York

Laudon, K. C., Laudon, J. P. 1991. Management information systems : A contemporary perspective. New York: Macmillan.

Laudon, C. K., Laudon, J. P. 2002. Management Information Systems. 7th Edition, Prentice Hall, New Jersey

Long, L. 1989. Management Information Systems. Prentice Hall, U.S.A.

Miller, J. G., & Sprague, L. G. 1975. Behind the growth in materials requirements planning. Harvard Business Review, 53 (5), 83-91

Özkök, B. 1997. İşletmelerde MRP II Uygulamaları. Mühendis ve Makine, Cilt: 38, Sayı: 454, Kasım

Pan, C. H. 1994. Sensitivity analysis of dynamic lot - sizing heuristics. Omega 22:251–61

Saydam, C., Evans J. R. 1990. A comparative performance analysis of the Wagner–Whitin algorithm and lot - sizing heuristics. Computers & Industrial Engineering 18:91–3.

Silver, E.A., Meal, H. C. 1973. Aheuristic for selecting lot size quantities for the case of a deterministic time varying demand rate and discrete opportunities for replenishment. Production and Inventory Management, 14:64–74.

Stair, R. M. 1992. Principles of Information Systems. Body & Fraser Publishing Company, Boston

Şahin, M. A. 2003. Plastik Ambalaj Malzemeleri Üreten Bir Firmada Malzeme İhtiyaç Planlama Uygulaması. Endüstri Mühendisliği Uygulaması. Selçuk Üniversitesi Endüstri Mühendisliği Bölümü

Şenel, N. 1996. MRP'de optimum Ürün Karışımının Belirlenmesi. Yüksek Lisans Tezi Gazi Üniversitesi Fen bilimleri Enstitüsü Endüstri Mühendisliği, Ankara

Ülgen, H. 1980. İşletme Yönetiminde Bilgisayar. Beta Yayınları, İstanbul

Wagner, H. M., Whitin, T. M. 1958. Dynamic version of the economic lot size model. Management Science, 5:89–96.

Yegül, F. 2002. ERP Kurumsal Kaynak Planlama. Yüksek Lisans Semineri, Endüstri Mühendisliği Ana Bilim Dalı, Gazi Üniversitesi, Ankara

Yenersoy, G. 1990. Malzeme Yönetim Sistemleri. No: 1, MA-PA Yayınları, İstanbul

Yozgat, U. 1998. Yönetim Bilişim Sistemleri (Management Information Systems), Beta, İstanbul

7. EK-A VISUAL BASIC PROGRAM KODLARI

```
Dim S, C, STM, U, EOQ, yıl_sip_say As Integer
Private Sub Command1_Click()
MSFlexGrid1.Cols = 15
MSFlexGrid1.Rows = 6
MSFlexGrid1.ColWidth(0) = "1600"
MSFlexGrid1.ColWidth(1) = "525"
MSFlexGrid1.ColWidth(2) = "525"
MSFlexGrid1.ColWidth(3) = "525"
MSFlexGrid1.ColWidth(4) = "525"
MSFlexGrid1.ColWidth(5) = "525"
MSFlexGrid1.ColWidth(6) = "525"
MSFlexGrid1.ColWidth(7) = "525"
MSFlexGrid1.ColWidth(8) = "525"
MSFlexGrid1.ColWidth(9) = "525"
MSFlexGrid1.ColWidth(10) = "525"
MSFlexGrid1.ColWidth(11) = "525"
MSFlexGrid1.ColWidth(12) = "525"
MSFlexGrid1.ColWidth(13) = "525"
MSFlexGrid1.ColWidth(14) = "700"
MSFlexGrid1.Width = "9200"
MSFlexGrid1.ColAlignment(14) = 6
MSFlexGrid1.TextMatrix(5, 0) = "Toplam Maliyet"
MSFlexGrid1.TextMatrix(0, 0) = "Aylar"
MSFlexGrid1.TextMatrix(1, 0) = "Net İhtiyaç"
MSFlexGrid1.TextMatrix(2, 0) = "Verilen Sipariş"
MSFlexGrid1.TextMatrix(3, 0) = "Stok Taşıma Maliyeti"
MSFlexGrid1.TextMatrix(4, 0) = "Hazırlık Maliyeti"
MSFlexGrid1.TextMatrix(0, 14) = "Toplam"
Dim i, j
i = MSFlexGrid1.Row
j = MSFlexGrid1.Col
For j = 1 To 13
MSFlexGrid1.TextMatrix(0, j) = j - 1
Next
For k = 1 To 12
b = Val(InputBox(k & ".ayın net ihtiyacını giriniz."))
MSFlexGrid1.TextMatrix(1, k + 1) = b
toplam = toplam + b
MSFlexGrid1.TextMatrix(1, 14) = toplam
Next
hm = Val(InputBox("Hazırlık maliyetini giriniz."))
For j = 1 To 12
MSFlexGrid1.TextMatrix(2, j) = MSFlexGrid1.TextMatrix(1, j + 1)
```

```
Next
For j = 1 To 13
MSFlexGrid1.TextMatrix(3, j) = 0
Next
For j = 1 To 13
MSFlexGrid1.TextMatrix(4, j) = Val(hm)
Next
For j = 1 To 12
D = Val(MSFlexGrid1.TextMatrix(2, j))
top_sip = top_sip + D
MSFlexGrid1.TextMatrix(2, 14) = top_sip
Next
For j = 1 To 12
e = Val(MSFlexGrid1.TextMatrix(3, j))
top_stk_tsm = top_stk_tsm + e
MSFlexGrid1.TextMatrix(3, 14) = top_stk_tsm
Next
For j = 1 To 12
f = Val(MSFlexGrid1.TextMatrix(4, j))
top_hm = top_hm + f
MSFlexGrid1.TextMatrix(4, 14) = top_hm
Next
MSFlexGrid1.TextMatrix(5, 14) = Val(MSFlexGrid1.TextMatrix(3, 14)) +
Val(MSFlexGrid1.TextMatrix(4, 14))
End Sub

Private Sub Command10_Click()
MSFlexGrid1.Cols = 13
MSFlexGrid1.Rows = 43
MSFlexGrid1.Width = "9000"
MSFlexGrid1.Height = "6000"
MSFlexGrid1.Top = "0"
MSFlexGrid1.ColWidth(1) = "525"
MSFlexGrid1.ColWidth(2) = "525"
MSFlexGrid1.ColWidth(3) = "525"
MSFlexGrid1.ColWidth(4) = "525"
MSFlexGrid1.ColWidth(5) = "525"
MSFlexGrid1.ColWidth(6) = "525"
MSFlexGrid1.ColWidth(7) = "525"
MSFlexGrid1.ColWidth(8) = "525"
MSFlexGrid1.ColWidth(9) = "525"
MSFlexGrid1.ColWidth(10) = "525"
MSFlexGrid1.ColWidth(11) = "525"
MSFlexGrid1.ColWidth(12) = "525"
MSFlexGrid1.ColWidth(0) = "2250"
MSFlexGrid1.TextMatrix(0, 0) = "Aylar"
MSFlexGrid1.TextMatrix(17, 0) = "Aylar"
MSFlexGrid1.TextMatrix(30, 0) = "Aylar"
```

```
MSFlexGrid1.TextMatrix(1, 0) = "Talep"
MSFlexGrid1.TextMatrix(2, 0) = "Hiz. Sipariş Maliyeti"
MSFlexGrid1.TextMatrix(3, 0) = "Stok Taşıma Maliyeti"
MSFlexGrid1.TextMatrix(4, 0) = "Stokta Taşınan Dönem Sayısı"
Dim i, j
i = MSFlexGrid1.Row
j = MSFlexGrid1.Col
Dim x, y
x = MSFlexGrid2.Row
y = MSFlexGrid2.Col
MSFlexGrid2.Cols = 14
MSFlexGrid2.Rows = 6
MSFlexGrid2.Top = "6250"
For j = 1 To 12
MSFlexGrid1.TextMatrix(0, j) = j
Next
hm = Val(InputBox("Hazırlık maliyetini giriniz."))
For k = 1 To 12
b = Val(InputBox(k & ".ayın talep değerini giriniz."))
MSFlexGrid1.TextMatrix(1, k) = b
MSFlexGrid2.TextMatrix(1, k) = b
toplam = toplam + b
MSFlexGrid2.TextMatrix(1, 13) = toplam
Next
For j = 1 To 12
MSFlexGrid1.TextMatrix(2, j) = hm
Next
For j = 1 To 12
MSFlexGrid1.TextMatrix(3, j) = 1
Next
For j = 1 To 12
MSFlexGrid1.TextMatrix(4, j) = j - 1
Next
For j = 1 To 12
MSFlexGrid1.TextMatrix(j + 4, j) = hm
Next
For j = 2 To 12
MSFlexGrid1.TextMatrix(5, j) = Val(MSFlexGrid1.TextMatrix(1, j)) *
Val(MSFlexGrid1.TextMatrix(4, j))
Next
For j = 3 To 12
MSFlexGrid1.TextMatrix(6, j) = Val(MSFlexGrid1.TextMatrix(5, j)) -
Val(MSFlexGrid1.TextMatrix(1, j))
Next
For j = 4 To 12
MSFlexGrid1.TextMatrix(7, j) = Val(MSFlexGrid1.TextMatrix(6, j)) -
Val(MSFlexGrid1.TextMatrix(1, j))
Next
```

```
For j = 5 To 12
MSFlexGrid1.TextMatrix(8, j) = Val(MSFlexGrid1.TextMatrix(7, j)) -
Val(MSFlexGrid1.TextMatrix(1, j))
Next
For j = 6 To 12
MSFlexGrid1.TextMatrix(9, j) = Val(MSFlexGrid1.TextMatrix(8, j)) -
Val(MSFlexGrid1.TextMatrix(1, j))
Next
For j = 7 To 12
MSFlexGrid1.TextMatrix(10, j) = Val(MSFlexGrid1.TextMatrix(9, j)) -
Val(MSFlexGrid1.TextMatrix(1, j))
Next
For j = 8 To 12
MSFlexGrid1.TextMatrix(11, j) = Val(MSFlexGrid1.TextMatrix(10, j)) -
Val(MSFlexGrid1.TextMatrix(1, j))
Next
For j = 9 To 12
MSFlexGrid1.TextMatrix(12, j) = Val(MSFlexGrid1.TextMatrix(11, j)) -
Val(MSFlexGrid1.TextMatrix(1, j))
Next
For j = 10 To 12
MSFlexGrid1.TextMatrix(13, j) = Val(MSFlexGrid1.TextMatrix(12, j)) -
Val(MSFlexGrid1.TextMatrix(1, j))
Next
For j = 11 To 12
MSFlexGrid1.TextMatrix(14, j) = Val(MSFlexGrid1.TextMatrix(13, j)) -
Val(MSFlexGrid1.TextMatrix(1, j))
Next
MSFlexGrid1.TextMatrix(15, 12) = Val(MSFlexGrid1.TextMatrix(14, 12)) -
Val(MSFlexGrid1.TextMatrix(1, 12))
For j = 1 To 12
MSFlexGrid1.TextMatrix(17, j) = j
Next
For j = 1 To 12
MSFlexGrid1.TextMatrix(30, j) = j
Next
For j = 1 To 12
MSFlexGrid1.TextMatrix(j + 17, j) = hm
Next
For j = 2 To 12
MSFlexGrid1.TextMatrix(18, j) = Val(MSFlexGrid1.TextMatrix(18, j - 1)) +
Val(MSFlexGrid1.TextMatrix(5, j))
Next
For j = 3 To 12
MSFlexGrid1.TextMatrix(19, j) = Val(MSFlexGrid1.TextMatrix(19, j - 1)) +
Val(MSFlexGrid1.TextMatrix(6, j))
Next
For j = 4 To 12
```

```
MSFlexGrid1.TextMatrix(20, j) = Val(MSFlexGrid1.TextMatrix(20, j - 1)) +
Val(MSFlexGrid1.TextMatrix(7, j))
Next
For j = 5 To 12
MSFlexGrid1.TextMatrix(21, j) = Val(MSFlexGrid1.TextMatrix(21, j - 1)) +
Val(MSFlexGrid1.TextMatrix(8, j))
Next
For j = 6 To 12
MSFlexGrid1.TextMatrix(22, j) = Val(MSFlexGrid1.TextMatrix(22, j - 1)) +
Val(MSFlexGrid1.TextMatrix(9, j))
Next
For j = 7 To 12
MSFlexGrid1.TextMatrix(23, j) = Val(MSFlexGrid1.TextMatrix(23, j - 1)) +
Val(MSFlexGrid1.TextMatrix(10, j))
Next
For j = 8 To 12
MSFlexGrid1.TextMatrix(24, j) = Val(MSFlexGrid1.TextMatrix(24, j - 1)) +
Val(MSFlexGrid1.TextMatrix(11, j))
Next
For j = 9 To 12
MSFlexGrid1.TextMatrix(25, j) = Val(MSFlexGrid1.TextMatrix(25, j - 1)) +
Val(MSFlexGrid1.TextMatrix(12, j))
Next
For j = 10 To 12
MSFlexGrid1.TextMatrix(26, j) = Val(MSFlexGrid1.TextMatrix(26, j - 1)) +
Val(MSFlexGrid1.TextMatrix(13, j))
Next
For j = 11 To 12
MSFlexGrid1.TextMatrix(27, j) = Val(MSFlexGrid1.TextMatrix(27, j - 1)) +
Val(MSFlexGrid1.TextMatrix(14, j))
Next
MSFlexGrid1.TextMatrix(28, 12) = Val(MSFlexGrid1.TextMatrix(28, 11)) +
Val(MSFlexGrid1.TextMatrix(15, 12))
For j = 1 To 12
MSFlexGrid1.TextMatrix(31, j) = Val(MSFlexGrid1.TextMatrix(18, j))
Next
For j = 2 To 12
MSFlexGrid1.TextMatrix(j + 30, j) = Val(MSFlexGrid1.TextMatrix(j + 29, j - 1)) +
Val(MSFlexGrid1.TextMatrix(5, j - 1))
Next
For j = 3 To 12
MSFlexGrid1.TextMatrix(32, j) = Val(MSFlexGrid1.TextMatrix(32, j - 1)) +
Val(MSFlexGrid1.TextMatrix(6, j))
Next
For j = 4 To 12
MSFlexGrid1.TextMatrix(33, j) = Val(MSFlexGrid1.TextMatrix(33, j - 1)) +
Val(MSFlexGrid1.TextMatrix(7, j))
Next
```

```
For j = 5 To 12
MSFlexGrid1.TextMatrix(34, j) = Val(MSFlexGrid1.TextMatrix(34, j - 1)) +
Val(MSFlexGrid1.TextMatrix(8, j))
Next
For j = 6 To 12
MSFlexGrid1.TextMatrix(35, j) = Val(MSFlexGrid1.TextMatrix(35, j - 1)) +
Val(MSFlexGrid1.TextMatrix(9, j))
Next
For j = 7 To 12
MSFlexGrid1.TextMatrix(36, j) = Val(MSFlexGrid1.TextMatrix(36, j - 1)) +
Val(MSFlexGrid1.TextMatrix(10, j))
Next
For j = 8 To 12
MSFlexGrid1.TextMatrix(37, j) = Val(MSFlexGrid1.TextMatrix(37, j - 1)) +
Val(MSFlexGrid1.TextMatrix(11, j))
Next
For j = 9 To 12
MSFlexGrid1.TextMatrix(38, j) = Val(MSFlexGrid1.TextMatrix(38, j - 1)) +
Val(MSFlexGrid1.TextMatrix(12, j))
Next
For j = 10 To 12
MSFlexGrid1.TextMatrix(39, j) = Val(MSFlexGrid1.TextMatrix(39, j - 1)) +
Val(MSFlexGrid1.TextMatrix(13, j))
Next
For j = 11 To 12
MSFlexGrid1.TextMatrix(40, j) = Val(MSFlexGrid1.TextMatrix(40, j - 1)) +
Val(MSFlexGrid1.TextMatrix(14, j))
Next
MSFlexGrid1.TextMatrix(41, 12) = Val(MSFlexGrid1.TextMatrix(41, 11)) +
Val(MSFlexGrid1.TextMatrix(15, 12))
MSFlexGrid2.Visible = True
MSFlexGrid2.ColWidth(0) = "1600"
MSFlexGrid2.ColWidth(1) = "525"
MSFlexGrid2.ColWidth(2) = "525"
MSFlexGrid2.ColWidth(3) = "525"
MSFlexGrid2.ColWidth(4) = "525"
MSFlexGrid2.ColWidth(5) = "525"
MSFlexGrid2.ColWidth(6) = "525"
MSFlexGrid2.ColWidth(7) = "525"
MSFlexGrid2.ColWidth(8) = "525"
MSFlexGrid2.ColWidth(9) = "525"
MSFlexGrid2.ColWidth(10) = "525"
MSFlexGrid2.ColWidth(11) = "525"
MSFlexGrid2.ColWidth(12) = "525"
MSFlexGrid2.ColWidth(13) = "700"
MSFlexGrid2.Width = "8750"
MSFlexGrid2.ColAlignment(13) = 6
MSFlexGrid2.TextMatrix(5, 0) = "Toplam Maliyet"
```

```
MSFlexGrid2.TextMatrix(0, 0) = "Aylar"
MSFlexGrid2.TextMatrix(1, 0) = "Net İhtiyaç"
MSFlexGrid2.TextMatrix(2, 0) = "Verilen Sipariş"
MSFlexGrid2.TextMatrix(3, 0) = "Stok Taşıma Maliyeti"
MSFlexGrid2.TextMatrix(4, 0) = "Hazırlık Maliyeti"
MSFlexGrid2.TextMatrix(0, 13) = "Toplam"
For y = 1 To 12
MSFlexGrid2.TextMatrix(0, y) = y
Next
MSFlexGrid2.TextMatrix(2, 1) = Val(MSFlexGrid2.TextMatrix(1, 1)) +
Val(MSFlexGrid2.TextMatrix(1, 2)) + Val(MSFlexGrid2.TextMatrix(1, 3))
MSFlexGrid2.TextMatrix(2, 4) = Val(MSFlexGrid2.TextMatrix(1, 4)) +
Val(MSFlexGrid2.TextMatrix(1, 5)) + Val(MSFlexGrid2.TextMatrix(1, 6))
For x = 7 To 12
C = Val(MSFlexGrid2.TextMatrix(1, x))
topl = topl + C
MSFlexGrid2.TextMatrix(2, 7) = topl
Next
For y = 1 To 11
If Val((MSFlexGrid2.TextMatrix(3, y))) >= Val((MSFlexGrid2.TextMatrix(1, y)))
Then
MSFlexGrid2.TextMatrix(3, y + 1) = Val(MSFlexGrid2.TextMatrix(3, y)) -
Val(MSFlexGrid2.TextMatrix(1, y))
Else
MSFlexGrid2.TextMatrix(3, y + 1) = Val((MSFlexGrid2.TextMatrix(2, y))) -
Val((MSFlexGrid2.TextMatrix(1, y)))
End If
Next
For y = 1 To 12
If Val(MSFlexGrid2.TextMatrix(2, y)) <> 0 Then
MSFlexGrid2.TextMatrix(4, y) = Val(S)
End If
Next
For y = 1 To 12
D = Val(MSFlexGrid2.TextMatrix(2, y))
top_sip = top_sip + D
MSFlexGrid2.TextMatrix(2, 13) = top_sip
Next
For y = 1 To 12
e = Val(MSFlexGrid2.TextMatrix(3, y))
top_stk_tsm = top_stk_tsm + e
MSFlexGrid2.TextMatrix(3, 13) = top_stk_tsm
Next
For y = 1 To 12
f = Val(MSFlexGrid2.TextMatrix(4, y))
top_hm = top_hm + f
MSFlexGrid2.TextMatrix(4, 13) = top_hm
Next
```

```
MSFlexGrid2.TextMatrix(5, 13) = Val(MSFlexGrid2.TextMatrix(3, 13)) +
Val(MSFlexGrid2.TextMatrix(4, 13))
End Sub
Private Sub Command2_Click()
hm = Val(InputBox("Hazırlık maliyetini giriniz."))
C = Val(InputBox("Birim maliyeti giriniz."))
h = Val(InputBox("Envanter taşıma maliyetini giriniz."))
MSFlexGrid1.Cols = 14
MSFlexGrid1.Rows = 6
MSFlexGrid1.ColWidth(0) = "1600"
MSFlexGrid1.ColWidth(1) = "525"
MSFlexGrid1.ColWidth(2) = "525"
MSFlexGrid1.ColWidth(3) = "525"
MSFlexGrid1.ColWidth(4) = "525"
MSFlexGrid1.ColWidth(5) = "525"
MSFlexGrid1.ColWidth(6) = "525"
MSFlexGrid1.ColWidth(7) = "525"
MSFlexGrid1.ColWidth(8) = "525"
MSFlexGrid1.ColWidth(9) = "525"
MSFlexGrid1.ColWidth(10) = "525"
MSFlexGrid1.ColWidth(11) = "525"
MSFlexGrid1.ColWidth(12) = "525"
MSFlexGrid1.ColWidth(13) = "700"
MSFlexGrid1.Width = "8750"
MSFlexGrid1.ColAlignment(13) = 6
MSFlexGrid1.TextMatrix(5, 0) = "Toplam Maliyet"
MSFlexGrid1.TextMatrix(0, 0) = "Aylar"
MSFlexGrid1.TextMatrix(1, 0) = "Net İhtiyaç"
MSFlexGrid1.TextMatrix(2, 0) = "Verilen Sipariş"
MSFlexGrid1.TextMatrix(3, 0) = "Stok Taşıma Maliyeti"
MSFlexGrid1.TextMatrix(4, 0) = "Hazırlık Maliyeti"
MSFlexGrid1.TextMatrix(0, 13) = "Toplam"
Dim i, j
i = MSFlexGrid1.Row
j = MSFlexGrid1.Col
For j = 1 To 12
MSFlexGrid1.TextMatrix(0, j) = j
Next
For k = 1 To 12
b = Val(InputBox(k & ".ayın net ihtiyacını giriniz."))
MSFlexGrid1.TextMatrix(1, k) = b
toplam = toplam + b
MSFlexGrid1.TextMatrix(1, 13) = toplam
Next
MSFlexGrid1.TextMatrix(4, 1) = hm
MSFlexGrid1.TextMatrix(4, 3) = hm
MSFlexGrid1.TextMatrix(4, 4) = hm
MSFlexGrid1.TextMatrix(2, 3) = MSFlexGrid1.TextMatrix(1, 3)
```

```
For j = 6 To 12
MSFlexGrid1.TextMatrix(4, j) = hm
MSFlexGrid1.TextMatrix(2, j) = MSFlexGrid1.TextMatrix(1, j)
MSFlexGrid1.TextMatrix(3, j) = 0
MSFlexGrid1.TextMatrix(4, j) = hm
Next
MSFlexGrid1.TextMatrix(2, 1) = Val(MSFlexGrid1.TextMatrix(1, 1)) +
Val(MSFlexGrid1.TextMatrix(1, 2))
MSFlexGrid1.TextMatrix(2, 4) = Val(MSFlexGrid1.TextMatrix(1, 4)) +
Val(MSFlexGrid1.TextMatrix(1, 5))
MSFlexGrid1.TextMatrix(3, 1) = 0
MSFlexGrid1.TextMatrix(3, 2) = MSFlexGrid1.TextMatrix(1, 2)
MSFlexGrid1.TextMatrix(3, 5) = MSFlexGrid1.TextMatrix(1, 5)
MSFlexGrid1.TextMatrix(3, 3) = 0
MSFlexGrid1.TextMatrix(3, 4) = 0
MSFlexGrid1.TextMatrix(4, 1) = 0
MSFlexGrid1.TextMatrix(3, 4) = hm
MSFlexGrid1.TextMatrix(4, 3) = hm
MSFlexGrid1.TextMatrix(4, 4) = hm
For j = 1 To 12
D = Val(MSFlexGrid1.TextMatrix(2, j))
top_sip = top_sip + D
MSFlexGrid1.TextMatrix(2, 13) = top_sip
Next
For j = 1 To 12
e = Val(MSFlexGrid1.TextMatrix(3, j))
top_stk_tsm = top_stk_tsm + e
MSFlexGrid1.TextMatrix(3, 13) = top_stk_tsm
Next
For j = 1 To 12
f = Val(MSFlexGrid1.TextMatrix(4, j))
top_hm = top_hm + f
MSFlexGrid1.TextMatrix(4, 13) = top_hm
Next
MSFlexGrid1.TextMatrix(5, 13) = Val(MSFlexGrid1.TextMatrix(3, 13)) +
Val(MSFlexGrid1.TextMatrix(4, 13))
End Sub

Private Sub Command3_Click()
MSFlexGrid2.Visible = True
MSFlexGrid2.Top = "6250"
MSFlexGrid1.Cols = 8
MSFlexGrid1.Rows = 14
hm = Val(InputBox("Hazırlık maliyetini giriniz."))
MSFlexGrid1.ColAlignment(0) = 0
MSFlexGrid1.ColWidth(0) = "600"
MSFlexGrid1.ColWidth(1) = "900"
MSFlexGrid1.ColWidth(2) = "1350"
```

```
MSFlexGrid1.ColWidth(3) = "1400"
MSFlexGrid1.ColWidth(4) = "1450"
MSFlexGrid1.ColWidth(5) = "1300"
MSFlexGrid1.ColWidth(6) = "1550"
MSFlexGrid1.ColWidth(7) = "650"
MSFlexGrid1.Width = "9400"
MSFlexGrid1.Height = "5250"
MSFlexGrid1.TextMatrix(0, 0) = "Dönem"
MSFlexGrid1.TextMatrix(0, 1) = "Net İhtiyaç"
MSFlexGrid1.TextMatrix(0, 2) = "Stokta Taş. Dön."
MSFlexGrid1.TextMatrix(0, 3) = "Muht. Sip. Mikt.(1)"
MSFlexGrid1.TextMatrix(0, 4) = "Taşıma Parti İçin(2)"
MSFlexGrid1.TextMatrix(0, 5) = "Mal. Bir. İçin(2/1)"
MSFlexGrid1.TextMatrix(0, 6) = "Bir. Haz. Mal.(Hm/1)"
MSFlexGrid1.TextMatrix(0, 7) = "Bir. Mal."
MSFlexGrid1.TextMatrix(1, 0) = "1"
MSFlexGrid1.TextMatrix(2, 0) = "2"
MSFlexGrid1.TextMatrix(3, 0) = "3"
MSFlexGrid1.TextMatrix(4, 0) = "2"
MSFlexGrid1.TextMatrix(5, 0) = "3"
MSFlexGrid1.TextMatrix(6, 0) = "2"
MSFlexGrid1.TextMatrix(7, 0) = "3"
MSFlexGrid1.TextMatrix(8, 0) = "4"
MSFlexGrid1.TextMatrix(9, 0) = "2"
MSFlexGrid1.TextMatrix(10, 0) = "3"
MSFlexGrid1.TextMatrix(11, 0) = "2"
MSFlexGrid1.TextMatrix(12, 0) = "3"
MSFlexGrid2.Cols = 14
MSFlexGrid2.Rows = 6
Dim i, j
i = MSFlexGrid1.Row
j = MSFlexGrid1.Col
x = MSFlexGrid2.Row
y = MSFlexGrid2.Col
For k = 1 To 12
b = Val(InputBox(k & ".ayın net ihtiyacını giriniz."))
MSFlexGrid1.TextMatrix(k, 1) = b
MSFlexGrid2.TextMatrix(1, k) = b
toplam = toplam + b
Next
MSFlexGrid1.AddItem "", 4
MSFlexGrid1.AddItem "", 7
MSFlexGrid1.AddItem "", 11
MSFlexGrid1.AddItem "", 14
MSFlexGrid1.AddItem "", 17
MSFlexGrid1.TextMatrix(4, 0) = 1
MSFlexGrid1.TextMatrix(4, 1) = MSFlexGrid1.TextMatrix(3, 1)
MSFlexGrid1.TextMatrix(7, 0) = 1
```

```
MSFlexGrid1.TextMatrix(7, 1) = MSFlexGrid1.TextMatrix(6, 1)
MSFlexGrid1.TextMatrix(11, 0) = 1
MSFlexGrid1.TextMatrix(11, 1) = MSFlexGrid1.TextMatrix(10, 1)
MSFlexGrid1.TextMatrix(14, 0) = 1
MSFlexGrid1.TextMatrix(14, 1) = MSFlexGrid1.TextMatrix(13, 1)
MSFlexGrid1.TextMatrix(17, 0) = 1
MSFlexGrid1.TextMatrix(17, 1) = MSFlexGrid1.TextMatrix(16, 1)
MSFlexGrid1.TextMatrix(1, 2) = 0
MSFlexGrid1.TextMatrix(2, 2) = 1
MSFlexGrid1.TextMatrix(3, 2) = 2
MSFlexGrid1.TextMatrix(4, 2) = 0
MSFlexGrid1.TextMatrix(5, 2) = 1
MSFlexGrid1.TextMatrix(6, 2) = 2
MSFlexGrid1.TextMatrix(7, 2) = 0
MSFlexGrid1.TextMatrix(8, 2) = 1
MSFlexGrid1.TextMatrix(9, 2) = 2
MSFlexGrid1.TextMatrix(10, 2) = 3
MSFlexGrid1.TextMatrix(11, 2) = 0
MSFlexGrid1.TextMatrix(12, 2) = 1
MSFlexGrid1.TextMatrix(13, 2) = 2
MSFlexGrid1.TextMatrix(14, 2) = 0
MSFlexGrid1.TextMatrix(15, 2) = 1
MSFlexGrid1.TextMatrix(16, 2) = 2
MSFlexGrid1.TextMatrix(17, 2) = 0
MSFlexGrid1.TextMatrix(1, 3) = MSFlexGrid1.TextMatrix(1, 1)
MSFlexGrid1.TextMatrix(4, 3) = MSFlexGrid1.TextMatrix(4, 1)
MSFlexGrid1.TextMatrix(7, 3) = MSFlexGrid1.TextMatrix(7, 1)
MSFlexGrid1.TextMatrix(11, 3) = MSFlexGrid1.TextMatrix(11, 1)
MSFlexGrid1.TextMatrix(14, 3) = MSFlexGrid1.TextMatrix(14, 1)
MSFlexGrid1.TextMatrix(17, 3) = MSFlexGrid1.TextMatrix(17, 1)
MSFlexGrid1.TextMatrix(2, 3) = Val(MSFlexGrid1.TextMatrix(1, 1)) +
Val(MSFlexGrid1.TextMatrix(2, 1))
MSFlexGrid1.TextMatrix(3, 3) = Val(MSFlexGrid1.TextMatrix(2, 3)) +
Val(MSFlexGrid1.TextMatrix(3, 1))
MSFlexGrid1.TextMatrix(5, 3) = Val(MSFlexGrid1.TextMatrix(4, 1)) +
Val(MSFlexGrid1.TextMatrix(5, 1))
MSFlexGrid1.TextMatrix(6, 3) = Val(MSFlexGrid1.TextMatrix(5, 3)) +
Val(MSFlexGrid1.TextMatrix(6, 1))
MSFlexGrid1.TextMatrix(8, 3) = Val(MSFlexGrid1.TextMatrix(7, 1)) +
Val(MSFlexGrid1.TextMatrix(8, 1))
MSFlexGrid1.TextMatrix(9, 3) = Val(MSFlexGrid1.TextMatrix(8, 3)) +
Val(MSFlexGrid1.TextMatrix(9, 1))
MSFlexGrid1.TextMatrix(10, 3) = Val(MSFlexGrid1.TextMatrix(9, 3)) +
Val(MSFlexGrid1.TextMatrix(10, 1))
MSFlexGrid1.TextMatrix(12, 3) = Val(MSFlexGrid1.TextMatrix(11, 1)) +
Val(MSFlexGrid1.TextMatrix(12, 1))
MSFlexGrid1.TextMatrix(13, 3) = Val(MSFlexGrid1.TextMatrix(12, 3)) +
Val(MSFlexGrid1.TextMatrix(13, 1))
```

```
MSFlexGrid1.TextMatrix(15, 3) = Val(MSFlexGrid1.TextMatrix(14, 1)) +
Val(MSFlexGrid1.TextMatrix(15, 1))
MSFlexGrid1.TextMatrix(16, 3) = Val(MSFlexGrid1.TextMatrix(15, 3)) +
Val(MSFlexGrid1.TextMatrix(16, 1))
MSFlexGrid1.TextMatrix(1, 4) = Val(MSFlexGrid1.TextMatrix(1, 2)) *
Val(MSFlexGrid1.TextMatrix(1, 1))
MSFlexGrid1.TextMatrix(4, 4) = Val(MSFlexGrid1.TextMatrix(4, 2)) *
Val(MSFlexGrid1.TextMatrix(4, 1))
MSFlexGrid1.TextMatrix(7, 4) = Val(MSFlexGrid1.TextMatrix(7, 2)) *
Val(MSFlexGrid1.TextMatrix(7, 1))
MSFlexGrid1.TextMatrix(11, 4) = Val(MSFlexGrid1.TextMatrix(11, 2)) *
Val(MSFlexGrid1.TextMatrix(11, 1))
MSFlexGrid1.TextMatrix(14, 4) = Val(MSFlexGrid1.TextMatrix(14, 2)) *
Val(MSFlexGrid1.TextMatrix(14, 1))
MSFlexGrid1.TextMatrix(17, 4) = Val(MSFlexGrid1.TextMatrix(17, 2)) *
Val(MSFlexGrid1.TextMatrix(17, 1))
MSFlexGrid1.TextMatrix(2, 4) = Val(MSFlexGrid1.TextMatrix(2, 1)) *
Val(MSFlexGrid1.TextMatrix(2, 2))
MSFlexGrid1.TextMatrix(3, 4) = Val(MSFlexGrid1.TextMatrix(3, 1)) *
Val(MSFlexGrid1.TextMatrix(3, 2))
MSFlexGrid1.TextMatrix(5, 4) = Val(MSFlexGrid1.TextMatrix(5, 1)) *
Val(MSFlexGrid1.TextMatrix(5, 2))
MSFlexGrid1.TextMatrix(6, 4) = Val(MSFlexGrid1.TextMatrix(6, 1)) *
Val(MSFlexGrid1.TextMatrix(6, 2))
MSFlexGrid1.TextMatrix(8, 4) = Val(MSFlexGrid1.TextMatrix(8, 1)) *
Val(MSFlexGrid1.TextMatrix(8, 2))
MSFlexGrid1.TextMatrix(9, 4) = Val(MSFlexGrid1.TextMatrix(9, 1)) *
Val(MSFlexGrid1.TextMatrix(9, 2))
MSFlexGrid1.TextMatrix(11, 4) = Val(MSFlexGrid1.TextMatrix(11, 1)) *
Val(MSFlexGrid1.TextMatrix(10, 2))
MSFlexGrid1.TextMatrix(12, 4) = Val(MSFlexGrid1.TextMatrix(12, 1)) *
Val(MSFlexGrid1.TextMatrix(12, 2))
MSFlexGrid1.TextMatrix(13, 4) = Val(MSFlexGrid1.TextMatrix(13, 1)) *
Val(MSFlexGrid1.TextMatrix(13, 2))
MSFlexGrid1.TextMatrix(15, 4) = Val(MSFlexGrid1.TextMatrix(15, 1)) *
Val(MSFlexGrid1.TextMatrix(15, 2))
MSFlexGrid1.TextMatrix(16, 4) = Val(MSFlexGrid1.TextMatrix(16, 1)) *
Val(MSFlexGrid1.TextMatrix(16, 2))
For i = 1 To 17
MSFlexGrid1.TextMatrix(i, 5) = Val(MSFlexGrid1.TextMatrix(i, 4)) /
Val(MSFlexGrid1.TextMatrix(i, 3))
Next
For i = 1 To 17
MSFlexGrid1.TextMatrix(i, 6) = Val(hm) / Val(MSFlexGrid1.TextMatrix(i, 3))
Next
For i = 1 To 17
MSFlexGrid1.TextMatrix(i, 7) = Val(MSFlexGrid1.TextMatrix(i, 5)) +
Val(MSFlexGrid1.TextMatrix(i, 6))
```

```
Next
For i = 1 To 17
MSFlexGrid1.TextMatrix(i, 5) = Round((MSFlexGrid1.TextMatrix(i, 5)), 2)
MSFlexGrid1.TextMatrix(i, 6) = Round((MSFlexGrid1.TextMatrix(i, 6)), 2)
Next
MSFlexGrid2.ColWidth(0) = "1600"
MSFlexGrid2.ColWidth(1) = "525"
MSFlexGrid2.ColWidth(2) = "525"
MSFlexGrid2.ColWidth(3) = "525"
MSFlexGrid2.ColWidth(4) = "525"
MSFlexGrid2.ColWidth(5) = "525"
MSFlexGrid2.ColWidth(6) = "525"
MSFlexGrid2.ColWidth(7) = "525"
MSFlexGrid2.ColWidth(8) = "525"
MSFlexGrid2.ColWidth(9) = "525"
MSFlexGrid2.ColWidth(10) = "525"
MSFlexGrid2.ColWidth(11) = "525"
MSFlexGrid2.ColWidth(12) = "525"
MSFlexGrid2.ColWidth(13) = "700"
MSFlexGrid2.Width = "8750"
MSFlexGrid2.ColAlignment(13) = 6
MSFlexGrid2.TextMatrix(1, 13) = toplam
MSFlexGrid2.TextMatrix(5, 0) = "Toplam Maliyet"
MSFlexGrid2.TextMatrix(0, 0) = "Aylar"
MSFlexGrid2.TextMatrix(1, 0) = "Net İhtiyaç"
MSFlexGrid2.TextMatrix(2, 0) = "Verilen Sipariş"
MSFlexGrid2.TextMatrix(3, 0) = "Stok Taşıma Maliyeti"
MSFlexGrid2.TextMatrix(4, 0) = "Hazırlık Maliyeti"
MSFlexGrid2.TextMatrix(0, 13) = "Toplam"
For y = 1 To 12
MSFlexGrid2.TextMatrix(0, y) = y
Next
MSFlexGrid2.TextMatrix(2, 1) = Val(MSFlexGrid2.TextMatrix(1, 1)) +
Val(MSFlexGrid2.TextMatrix(1, 2))
MSFlexGrid2.TextMatrix(2, 3) = Val(MSFlexGrid2.TextMatrix(1, 3)) +
Val(MSFlexGrid2.TextMatrix(1, 4))
MSFlexGrid2.TextMatrix(2, 5) = Val(MSFlexGrid2.TextMatrix(1, 5)) +
Val(MSFlexGrid2.TextMatrix(1, 6)) + Val(MSFlexGrid2.TextMatrix(1, 7))
MSFlexGrid2.TextMatrix(2, 8) = Val(MSFlexGrid2.TextMatrix(1, 8)) +
Val(MSFlexGrid2.TextMatrix(1, 9))
MSFlexGrid2.TextMatrix(2, 10) = Val(MSFlexGrid2.TextMatrix(1, 10)) +
Val(MSFlexGrid2.TextMatrix(1, 11))
MSFlexGrid2.TextMatrix(2, 12) = MSFlexGrid2.TextMatrix(1, 12)
For y = 1 To 11
If Val((MSFlexGrid2.TextMatrix(3, y))) >= Val((MSFlexGrid2.TextMatrix(1, y)))
Then
MSFlexGrid2.TextMatrix(3, y + 1) = Val(MSFlexGrid2.TextMatrix(3, y)) -
Val(MSFlexGrid2.TextMatrix(1, y))
```

```
Else
MSFlexGrid2.TextMatrix(3, y + 1) = Val(MSFlexGrid2.TextMatrix(2, y)) -
Val(MSFlexGrid2.TextMatrix(3, y)) - Val(MSFlexGrid2.TextMatrix(1, y))
End If
Next
For y = 1 To 12
If Val(MSFlexGrid2.TextMatrix(3, y)) = 0 Then
MSFlexGrid2.TextMatrix(4, y) = Val(hm)
End If
Next
For j = 1 To 12
D = Val(MSFlexGrid2.TextMatrix(2, j))
top_sip = top_sip + D
MSFlexGrid2.TextMatrix(2, 13) = top_sip
Next
For j = 1 To 12
e = Val(MSFlexGrid2.TextMatrix(3, j))
top_stk_tsm = top_stk_tsm + e
MSFlexGrid2.TextMatrix(3, 13) = top_stk_tsm
Next
For j = 1 To 12
f = Val(MSFlexGrid2.TextMatrix(4, j))
top_hm = top_hm + f
MSFlexGrid2.TextMatrix(4, 13) = top_hm
Next
MSFlexGrid2.TextMatrix(5, 13) = Val(MSFlexGrid2.TextMatrix(3, 13)) +
Val(MSFlexGrid2.TextMatrix(4, 13))
End Sub

Private Sub Command4_Click()
MSFlexGrid1.Cols = 14
MSFlexGrid1.Rows = 6
MSFlexGrid1.ColWidth(0) = "1600"
MSFlexGrid1.ColWidth(1) = "525"
MSFlexGrid1.ColWidth(2) = "525"
MSFlexGrid1.ColWidth(3) = "525"
MSFlexGrid1.ColWidth(4) = "525"
MSFlexGrid1.ColWidth(5) = "525"
MSFlexGrid1.ColWidth(6) = "525"
MSFlexGrid1.ColWidth(7) = "525"
MSFlexGrid1.ColWidth(8) = "525"
MSFlexGrid1.ColWidth(9) = "525"
MSFlexGrid1.ColWidth(10) = "525"
MSFlexGrid1.ColWidth(11) = "525"
MSFlexGrid1.ColWidth(12) = "525"
MSFlexGrid1.ColWidth(13) = "700"
MSFlexGrid1.Width = "8750"
MSFlexGrid1.ColAlignment(13) = 6
```

```
MSFlexGrid1.TextMatrix(5, 0) = "Toplam Maliyet"
MSFlexGrid1.TextMatrix(0, 0) = "Aylar"
MSFlexGrid1.TextMatrix(1, 0) = "Net İhtiyaç"
MSFlexGrid1.TextMatrix(2, 0) = "Verilen Sipariş"
MSFlexGrid1.TextMatrix(3, 0) = "Stok Taşıma Maliyeti"
MSFlexGrid1.TextMatrix(4, 0) = "Hazırlık Maliyeti"
MSFlexGrid1.TextMatrix(0, 13) = "Toplam"
Dim i, j
i = MSFlexGrid1.Row
j = MSFlexGrid1.Col
For j = 1 To 12
MSFlexGrid1.TextMatrix(0, j) = j
Next
For k = 1 To 12
b = Val(InputBox(k & ".ayın net ihtiyacını giriniz."))
MSFlexGrid1.TextMatrix(1, k) = b
toplam = toplam + b
MSFlexGrid1.TextMatrix(1, 13) = toplam
Next
hm = Val(InputBox("Hazırlık maliyetini giriniz."))
MSFlexGrid1.TextMatrix(2, 1) = hm
MSFlexGrid1.TextMatrix(3, 1) = 0
For j = 1 To 11
If Val((MSFlexGrid1.TextMatrix(3, j))) >= Val((MSFlexGrid1.TextMatrix(1, j)))
Then
MSFlexGrid1.TextMatrix(3, j + 1) = Val(MSFlexGrid1.TextMatrix(3, j)) -
Val(MSFlexGrid1.TextMatrix(1, j))
Else
MSFlexGrid1.TextMatrix(2, j) = Val(hm)
MSFlexGrid1.TextMatrix(3, j + 1) = Val(hm) + Val((MSFlexGrid1.TextMatrix(3,
j))) - Val((MSFlexGrid1.TextMatrix(1, j)))
End If
Next
For j = 1 To 12
MSFlexGrid1.TextMatrix(4, j) = MSFlexGrid1.TextMatrix(2, j)
Next
For j = 1 To 12
D = Val(MSFlexGrid1.TextMatrix(2, j))
top_sip = top_sip + D
MSFlexGrid1.TextMatrix(2, 13) = top_sip
Next
For j = 1 To 12
e = Val(MSFlexGrid1.TextMatrix(3, j))
top_stk_tsm = top_stk_tsm + e
MSFlexGrid1.TextMatrix(3, 13) = top_stk_tsm
Next
For j = 1 To 12
f = Val(MSFlexGrid1.TextMatrix(4, j))
```

```
top_hm = top_hm + f
MSFlexGrid1.TextMatrix(4, 13) = top_hm
Next
MSFlexGrid1.TextMatrix(5, 13) = Val(MSFlexGrid1.TextMatrix(3, 13)) +
Val(MSFlexGrid1.TextMatrix(4, 13))
End Sub

Private Sub Command5_Click()
MSFlexGrid1.Cols = 14
MSFlexGrid1.Rows = 6
MSFlexGrid1.ColWidth(0) = "1600"
MSFlexGrid1.ColWidth(1) = "525"
MSFlexGrid1.ColWidth(2) = "525"
MSFlexGrid1.ColWidth(3) = "525"
MSFlexGrid1.ColWidth(4) = "525"
MSFlexGrid1.ColWidth(5) = "525"
MSFlexGrid1.ColWidth(6) = "525"
MSFlexGrid1.ColWidth(7) = "525"
MSFlexGrid1.ColWidth(8) = "525"
MSFlexGrid1.ColWidth(9) = "525"
MSFlexGrid1.ColWidth(10) = "525"
MSFlexGrid1.ColWidth(11) = "525"
MSFlexGrid1.ColWidth(12) = "525"
MSFlexGrid1.ColWidth(13) = "700"
MSFlexGrid1.Width = "8750"
MSFlexGrid1.ColAlignment(13) = 6
MSFlexGrid1.TextMatrix(5, 0) = "Toplam Maliyet"
MSFlexGrid1.TextMatrix(0, 0) = "Aylar"
MSFlexGrid1.TextMatrix(1, 0) = "Net İhtiyaç"
MSFlexGrid1.TextMatrix(2, 0) = "Verilen Sipariş"
MSFlexGrid1.TextMatrix(3, 0) = "Stok Taşıma Maliyeti"
MSFlexGrid1.TextMatrix(4, 0) = "Hazırlık Maliyeti"
MSFlexGrid1.TextMatrix(0, 13) = "Toplam"
Dim i, j
i = MSFlexGrid1.Row
j = MSFlexGrid1.Col
For j = 1 To 12
MSFlexGrid1.TextMatrix(0, j) = j
Next
For k = 1 To 12
b = Val(InputBox(k & ".ayın net ihtiyacını giriniz."))
MSFlexGrid1.TextMatrix(1, k) = b
toplam = toplam + b
MSFlexGrid1.TextMatrix(1, 13) = toplam
Next
hm = Val(InputBox("Hazırlık maliyetini giriniz."))
MSFlexGrid1.TextMatrix(4, 1) = hm
MSFlexGrid1.TextMatrix(3, 1) = 0
```

```
MSFlexGrid1.TextMatrix(2, 1) = Val(MSFlexGrid1.TextMatrix(1, 1)) +
Val(MSFlexGrid1.TextMatrix(1, 2)) + Val(MSFlexGrid1.TextMatrix(1, 3))
MSFlexGrid1.TextMatrix(2, 4) = Val(MSFlexGrid1.TextMatrix(1, 4)) +
Val(MSFlexGrid1.TextMatrix(1, 5)) + Val(MSFlexGrid1.TextMatrix(1, 6))
MSFlexGrid1.TextMatrix(2, 7) = Val(MSFlexGrid1.TextMatrix(1, 7)) +
Val(MSFlexGrid1.TextMatrix(1, 8)) + Val(MSFlexGrid1.TextMatrix(1, 9))
MSFlexGrid1.TextMatrix(2, 10) = Val(MSFlexGrid1.TextMatrix(1, 10)) +
Val(MSFlexGrid1.TextMatrix(1, 11)) + Val(MSFlexGrid1.TextMatrix(1, 12))
MSFlexGrid1.TextMatrix(4, 1) = hm
MSFlexGrid1.TextMatrix(4, 4) = hm
MSFlexGrid1.TextMatrix(4, 7) = hm
MSFlexGrid1.TextMatrix(4, 10) = hm
MSFlexGrid1.TextMatrix(3, 1) = 0
For j = 1 To 11
If Val((MSFlexGrid1.TextMatrix(3, j))) >= Val((MSFlexGrid1.TextMatrix(1, j)))
Then
MSFlexGrid1.TextMatrix(3, j + 1) = Val(MSFlexGrid1.TextMatrix(3, j)) -
Val(MSFlexGrid1.TextMatrix(1, j))
Else
MSFlexGrid1.TextMatrix(3, j + 1) = Val((MSFlexGrid1.TextMatrix(2, j))) -
Val((MSFlexGrid1.TextMatrix(1, j)))
End If
Next
For j = 1 To 12
D = Val(MSFlexGrid1.TextMatrix(2, j))
top_sip = top_sip + D
MSFlexGrid1.TextMatrix(2, 13) = top_sip
Next
For j = 1 To 12
e = Val(MSFlexGrid1.TextMatrix(3, j))
top_stk_tsm = top_stk_tsm + e
MSFlexGrid1.TextMatrix(3, 13) = top_stk_tsm
Next
For j = 1 To 12
f = Val(MSFlexGrid1.TextMatrix(4, j))
top_hm = top_hm + f
MSFlexGrid1.TextMatrix(4, 13) = top_hm
Next
MSFlexGrid1.TextMatrix(5, 13) = Val(MSFlexGrid1.TextMatrix(3, 13)) +
Val(MSFlexGrid1.TextMatrix(4, 13))
End Sub

Private Sub Command6_Click()
S = InputBox("Hazırlık maliyetini giriniz")
STM = InputBox("Stok taşıma maliyetini giriniz")
U = InputBox("Yıllık kullanım miktarını giriniz")
EOQ = (((2 * U * S) / (STM)) ^ (1 / 2))
MSFlexGrid1.Cols = 14
```

```
MSFlexGrid1.Rows = 6
MSFlexGrid1.ColWidth(0) = "1600"
MSFlexGrid1.ColWidth(1) = "525"
MSFlexGrid1.ColWidth(2) = "525"
MSFlexGrid1.ColWidth(3) = "525"
MSFlexGrid1.ColWidth(4) = "525"
MSFlexGrid1.ColWidth(5) = "525"
MSFlexGrid1.ColWidth(6) = "525"
MSFlexGrid1.ColWidth(7) = "525"
MSFlexGrid1.ColWidth(8) = "525"
MSFlexGrid1.ColWidth(9) = "525"
MSFlexGrid1.ColWidth(10) = "525"
MSFlexGrid1.ColWidth(11) = "525"
MSFlexGrid1.ColWidth(12) = "525"
MSFlexGrid1.ColWidth(13) = "700"
MSFlexGrid1.Width = "8750"
MSFlexGrid1.ColAlignment(13) = 6
MSFlexGrid1.TextMatrix(5, 0) = "Toplam Maliyet"
MSFlexGrid1.TextMatrix(0, 0) = "Aylar"
MSFlexGrid1.TextMatrix(1, 0) = "Net İhtiyaç"
MSFlexGrid1.TextMatrix(2, 0) = "Verilen Sipariş"
MSFlexGrid1.TextMatrix(3, 0) = "Stok Taşıma Maliyeti"
MSFlexGrid1.TextMatrix(4, 0) = "Hazırlık Maliyeti"
MSFlexGrid1.TextMatrix(0, 13) = "Toplam"
Dim i, j
i = MSFlexGrid1.Row
j = MSFlexGrid1.Col
For j = 1 To 12
MSFlexGrid1.TextMatrix(0, j) = j
Next
For k = 1 To 12
b = Val(InputBox(k & ".ayın net ihtiyacını giriniz."))
MSFlexGrid1.TextMatrix(1, k) = b
toplam = toplam + b
MSFlexGrid1.TextMatrix(1, 13) = toplam
Next
For j = 1 To 11
If Val((MSFlexGrid1.TextMatrix(3, j))) >= Val((MSFlexGrid1.TextMatrix(1, j)))
Then
MSFlexGrid1.TextMatrix(3, j + 1) = Val(MSFlexGrid1.TextMatrix(3, j)) -
Val(MSFlexGrid1.TextMatrix(1, j))
Else
MSFlexGrid1.TextMatrix(2, j) = Val(EOQ)
MSFlexGrid1.TextMatrix(3, j + 1) = Val(S) + Val((MSFlexGrid1.TextMatrix(3, j)))
- Val((MSFlexGrid1.TextMatrix(1, j)))
End If
Next
For j = 1 To 12
```

```
If Val(MSFlexGrid1.TextMatrix(2, j)) <> 0 Then
MSFlexGrid1.TextMatrix(4, j) = Val(S)
End If
Next
For j = 1 To 12
D = Val(MSFlexGrid1.TextMatrix(2, j))
top_sip = top_sip + D
MSFlexGrid1.TextMatrix(2, 13) = top_sip
Next
For j = 1 To 12
e = Val(MSFlexGrid1.TextMatrix(3, j))
top_stk_tsm = top_stk_tsm + e
MSFlexGrid1.TextMatrix(3, 13) = top_stk_tsm
Next
For j = 1 To 12
f = Val(MSFlexGrid1.TextMatrix(4, j))
top_hm = top_hm + f
MSFlexGrid1.TextMatrix(4, 13) = top_hm
Next
MSFlexGrid1.TextMatrix(5, 13) = Val(MSFlexGrid1.TextMatrix(3, 13)) +
Val(MSFlexGrid1.TextMatrix(4, 13))
End Sub

Private Sub Command7_Click()
MSFlexGrid1.Cols = 14
MSFlexGrid1.Rows = 6
MSFlexGrid1.ColWidth(0) = "1600"
MSFlexGrid1.ColWidth(1) = "525"
MSFlexGrid1.ColWidth(2) = "525"
MSFlexGrid1.ColWidth(3) = "525"
MSFlexGrid1.ColWidth(4) = "525"
MSFlexGrid1.ColWidth(5) = "525"
MSFlexGrid1.ColWidth(6) = "525"
MSFlexGrid1.ColWidth(7) = "525"
MSFlexGrid1.ColWidth(8) = "525"
MSFlexGrid1.ColWidth(9) = "525"
MSFlexGrid1.ColWidth(10) = "525"
MSFlexGrid1.ColWidth(11) = "525"
MSFlexGrid1.ColWidth(12) = "525"
MSFlexGrid1.ColWidth(13) = "700"
MSFlexGrid1.ColAlignment(12) = 6
MSFlexGrid1.ColAlignment(13) = 6
MSFlexGrid1.Width = "8750"
MSFlexGrid1.TextMatrix(5, 0) = "Toplam Maliyet"
MSFlexGrid1.TextMatrix(0, 0) = "Aylar"
MSFlexGrid1.TextMatrix(1, 0) = "Net İhtiyaç"
MSFlexGrid1.TextMatrix(2, 0) = "Verilen Sipariş"
MSFlexGrid1.TextMatrix(3, 0) = "Stok Taşıma Maliyeti"
```

```
MSFlexGrid1.TextMatrix(4, 0) = "Hazırlık Maliyeti"
MSFlexGrid1.TextMatrix(0, 13) = "Toplam"
Dim i, j
i = MSFlexGrid1.Row
j = MSFlexGrid1.Col
For j = 1 To 12
MSFlexGrid1.TextMatrix(0, j) = j
Next
For k = 1 To 12
b = Val(InputBox(k & ".ayın net ihtiyacını giriniz."))
MSFlexGrid1.TextMatrix(1, k) = b
toplam = toplam + b
MSFlexGrid1.TextMatrix(1, 13) = toplam
Next
hm = Val(InputBox("Hazırlık maliyetini giriniz."))
MSFlexGrid1.TextMatrix(4, 1) = hm
MSFlexGrid1.TextMatrix(4, 3) = hm
MSFlexGrid1.TextMatrix(4, 5) = hm
MSFlexGrid1.TextMatrix(4, 7) = hm
MSFlexGrid1.TextMatrix(4, 9) = hm
MSFlexGrid1.TextMatrix(4, 11) = hm
MSFlexGrid1.TextMatrix(3, 1) = 0
MSFlexGrid1.TextMatrix(3, 3) = 0
MSFlexGrid1.TextMatrix(3, 5) = 0
MSFlexGrid1.TextMatrix(3, 7) = 0
MSFlexGrid1.TextMatrix(3, 9) = 0
MSFlexGrid1.TextMatrix(3, 11) = 0
MSFlexGrid1.TextMatrix(2, 1) = Val(MSFlexGrid1.TextMatrix(1, 1)) +
Val(MSFlexGrid1.TextMatrix(1, 2))
MSFlexGrid1.TextMatrix(2, 3) = Val(MSFlexGrid1.TextMatrix(1, 3)) +
Val(MSFlexGrid1.TextMatrix(1, 4))
MSFlexGrid1.TextMatrix(2, 5) = Val(MSFlexGrid1.TextMatrix(1, 5)) +
Val(MSFlexGrid1.TextMatrix(1, 6))
MSFlexGrid1.TextMatrix(2, 7) = Val(MSFlexGrid1.TextMatrix(1, 7)) +
Val(MSFlexGrid1.TextMatrix(1, 8))
MSFlexGrid1.TextMatrix(2, 9) = Val(MSFlexGrid1.TextMatrix(1, 9)) +
Val(MSFlexGrid1.TextMatrix(1, 10))
MSFlexGrid1.TextMatrix(2, 11) = Val(MSFlexGrid1.TextMatrix(1, 11)) +
Val(MSFlexGrid1.TextMatrix(1, 12))
MSFlexGrid1.TextMatrix(3, 2) = Val(MSFlexGrid1.TextMatrix(1, 2))
MSFlexGrid1.TextMatrix(3, 4) = Val(MSFlexGrid1.TextMatrix(1, 4))
MSFlexGrid1.TextMatrix(3, 6) = Val(MSFlexGrid1.TextMatrix(1, 6))
MSFlexGrid1.TextMatrix(3, 8) = Val(MSFlexGrid1.TextMatrix(1, 8))
MSFlexGrid1.TextMatrix(3, 10) = Val(MSFlexGrid1.TextMatrix(1, 10))
MSFlexGrid1.TextMatrix(3, 12) = Val(MSFlexGrid1.TextMatrix(1, 12))
C = Val(MSFlexGrid1.TextMatrix(1, 13))
yıl_sip_say = C / hm
sip_ver_ara = 12 / Val(yıl_sip_say)
```

```
Label6.Caption = Val(yıl_sip_say)
Label8.Caption = Val(sip_ver_ara)
For j = 1 To 6
For n = 1 To 6
j = (2 * n) - 1
MSFlexGrid1.TextMatrix(2, j) = Val(MSFlexGrid1.TextMatrix(1, ((2 * n) - 1))) +
Val(MSFlexGrid1.TextMatrix(1, (2 * n)))
Next
Next
For j = 1 To 12
D = Val(MSFlexGrid1.TextMatrix(2, j))
top_sip = top_sip + D
MSFlexGrid1.TextMatrix(2, 13) = top_sip
Next
For j = 1 To 12
e = Val(MSFlexGrid1.TextMatrix(3, j))
top_stk_tsm = top_stk_tsm + e
MSFlexGrid1.TextMatrix(3, 13) = top_stk_tsm
Next
For j = 1 To 12
f = Val(MSFlexGrid1.TextMatrix(4, j))
top_hm = top_hm + f
MSFlexGrid1.TextMatrix(4, 13) = top_hm
Next
MSFlexGrid1.TextMatrix(5, 13) = Val(MSFlexGrid1.TextMatrix(3, 13)) +
Val(MSFlexGrid1.TextMatrix(4, 13))
Label5.Visible = True
Label6.Visible = True
Label7.Visible = True
Label8.Visible = True
End Sub

Private Sub Command8_Click()
MSFlexGrid2.Visible = True
MSFlexGrid1.Cols = 5
MSFlexGrid1.Rows = 18
MSFlexGrid2.Top = "5250"
MSFlexGrid2.ColWidth(0) = "600"
MSFlexGrid1.ColWidth(0) = "650"
MSFlexGrid1.ColWidth(1) = "850"
MSFlexGrid1.ColWidth(2) = "2300"
MSFlexGrid1.ColWidth(3) = "2000"
MSFlexGrid1.ColWidth(4) = "1100"
MSFlexGrid1.ColAlignment(0) = 6
MSFlexGrid1.Width = "7500"
MSFlexGrid1.Height = "4500"
MSFlexGrid2.Width = "8750"
EPP = Ekonomik_parça_dönem_faktörü
```

```
S = InputBox("Hazırlık maliyetini giriniz")
Ip = InputBox("Dönemsel stok taşıma maliyetini giriniz")
C = InputBox("Birim maliyeti giriniz")
EPP = Val(S) / Val(Ip * C)
MSFlexGrid1.TextMatrix(0, 0) = "Dönem"
MSFlexGrid1.TextMatrix(0, 1) = "Net İhtiyaç"
MSFlexGrid1.TextMatrix(0, 2) = "Stokta Taşındığı Dönem Sayısı"
MSFlexGrid1.TextMatrix(0, 3) = "Muhtemel Parti Büyüklüğü"
MSFlexGrid1.TextMatrix(0, 4) = "Parça-Dönem"
MSFlexGrid1.TextMatrix(1, 0) = "1"
MSFlexGrid1.TextMatrix(2, 0) = "2"
MSFlexGrid1.TextMatrix(3, 0) = "3"
MSFlexGrid1.TextMatrix(4, 0) = "4"
MSFlexGrid1.TextMatrix(5, 0) = "2"
MSFlexGrid1.TextMatrix(6, 0) = "3"
MSFlexGrid1.TextMatrix(7, 0) = "2"
MSFlexGrid1.TextMatrix(8, 0) = "3"
MSFlexGrid1.TextMatrix(9, 0) = "2"
MSFlexGrid1.TextMatrix(10, 0) = "3"
MSFlexGrid1.TextMatrix(11, 0) = "2"
MSFlexGrid1.TextMatrix(12, 0) = "3"
MSFlexGrid2.Cols = 14
MSFlexGrid2.Rows = 6
Dim i, j
i = MSFlexGrid1.Row
j = MSFlexGrid1.Col
x = MSFlexGrid2.Row
y = MSFlexGrid2.Col
MSFlexGrid2.ColAlignment(13) = 6
For k = 1 To 12
b = Val(InputBox(k & ".ayın net ihtiyacını giriniz."))
MSFlexGrid1.TextMatrix(k, 1) = b
MSFlexGrid2.TextMatrix(1, k) = b
toplam = toplam + b
Next
MSFlexGrid1.AddItem "", 5
MSFlexGrid1.AddItem "", 8
MSFlexGrid1.AddItem "", 11
MSFlexGrid1.AddItem "", 14
MSFlexGrid1.AddItem "", 17
MSFlexGrid1.TextMatrix(5, 0) = 1
MSFlexGrid1.TextMatrix(5, 1) = MSFlexGrid1.TextMatrix(4, 1)
MSFlexGrid1.TextMatrix(8, 0) = 1
MSFlexGrid1.TextMatrix(8, 1) = MSFlexGrid1.TextMatrix(7, 1)
MSFlexGrid1.TextMatrix(11, 0) = 1
MSFlexGrid1.TextMatrix(11, 1) = MSFlexGrid1.TextMatrix(10, 1)
MSFlexGrid1.TextMatrix(14, 0) = 1
MSFlexGrid1.TextMatrix(14, 1) = MSFlexGrid1.TextMatrix(13, 1)
```

```
MSFlexGrid1.TextMatrix(17, 0) = 1
MSFlexGrid1.TextMatrix(17, 1) = MSFlexGrid1.TextMatrix(16, 1)
MSFlexGrid1.TextMatrix(1, 2) = 0
MSFlexGrid1.TextMatrix(2, 2) = 1
MSFlexGrid1.TextMatrix(3, 2) = 2
MSFlexGrid1.TextMatrix(4, 2) = 3
MSFlexGrid1.TextMatrix(5, 2) = 0
MSFlexGrid1.TextMatrix(6, 2) = 1
MSFlexGrid1.TextMatrix(7, 2) = 2
MSFlexGrid1.TextMatrix(8, 2) = 0
MSFlexGrid1.TextMatrix(9, 2) = 1
MSFlexGrid1.TextMatrix(10, 2) = 2
MSFlexGrid1.TextMatrix(11, 2) = 0
MSFlexGrid1.TextMatrix(12, 2) = 1
MSFlexGrid1.TextMatrix(13, 2) = 2
MSFlexGrid1.TextMatrix(14, 2) = 0
MSFlexGrid1.TextMatrix(15, 2) = 1
MSFlexGrid1.TextMatrix(16, 2) = 2
MSFlexGrid1.TextMatrix(17, 2) = 0
MSFlexGrid1.TextMatrix(1, 3) = MSFlexGrid1.TextMatrix(1, 1)
MSFlexGrid1.TextMatrix(2, 3) = Val(MSFlexGrid1.TextMatrix(1, 3)) +
Val(MSFlexGrid1.TextMatrix(2, 1))
MSFlexGrid1.TextMatrix(3, 3) = Val(MSFlexGrid1.TextMatrix(2, 3)) +
Val(MSFlexGrid1.TextMatrix(3, 1))
MSFlexGrid1.TextMatrix(4, 3) = Val(MSFlexGrid1.TextMatrix(3, 3)) +
Val(MSFlexGrid1.TextMatrix(4, 1))
MSFlexGrid1.TextMatrix(5, 3) = MSFlexGrid1.TextMatrix(5, 1)
MSFlexGrid1.TextMatrix(6, 3) = Val(MSFlexGrid1.TextMatrix(5, 3)) +
Val(MSFlexGrid1.TextMatrix(6, 1))
MSFlexGrid1.TextMatrix(7, 3) = Val(MSFlexGrid1.TextMatrix(6, 3)) +
Val(MSFlexGrid1.TextMatrix(7, 1))
MSFlexGrid1.TextMatrix(8, 3) = MSFlexGrid1.TextMatrix(8, 1)
MSFlexGrid1.TextMatrix(9, 3) = Val(MSFlexGrid1.TextMatrix(8, 3)) +
Val(MSFlexGrid1.TextMatrix(9, 1))
MSFlexGrid1.TextMatrix(10, 3) = Val(MSFlexGrid1.TextMatrix(9, 3)) +
Val(MSFlexGrid1.TextMatrix(10, 1))
MSFlexGrid1.TextMatrix(11, 3) = MSFlexGrid1.TextMatrix(11, 1)
MSFlexGrid1.TextMatrix(12, 3) = Val(MSFlexGrid1.TextMatrix(11, 3)) +
Val(MSFlexGrid1.TextMatrix(12, 1))
MSFlexGrid1.TextMatrix(13, 3) = Val(MSFlexGrid1.TextMatrix(12, 3)) +
Val(MSFlexGrid1.TextMatrix(13, 1))
MSFlexGrid1.TextMatrix(14, 3) = MSFlexGrid1.TextMatrix(14, 1)
MSFlexGrid1.TextMatrix(15, 3) = Val(MSFlexGrid1.TextMatrix(14, 3)) +
Val(MSFlexGrid1.TextMatrix(15, 1))
MSFlexGrid1.TextMatrix(16, 3) = Val(MSFlexGrid1.TextMatrix(15, 3)) +
Val(MSFlexGrid1.TextMatrix(16, 1))
MSFlexGrid1.TextMatrix(17, 3) = MSFlexGrid1.TextMatrix(17, 1)
```

```
MSFlexGrid1.TextMatrix(1, 4) = Val(MSFlexGrid1.TextMatrix(1, 1)) *
Val(MSFlexGrid1.TextMatrix(1, 2))
MSFlexGrid1.TextMatrix(2, 4) = (Val(MSFlexGrid1.TextMatrix(2, 1)) *
Val(MSFlexGrid1.TextMatrix(2, 2))) + Val(MSFlexGrid1.TextMatrix(1, 4))
MSFlexGrid1.TextMatrix(3, 4) = (Val(MSFlexGrid1.TextMatrix(3, 1)) *
Val(MSFlexGrid1.TextMatrix(3, 2))) + Val(MSFlexGrid1.TextMatrix(2, 4))
MSFlexGrid1.TextMatrix(4, 4) = (Val(MSFlexGrid1.TextMatrix(4, 1)) *
Val(MSFlexGrid1.TextMatrix(4, 2))) + Val(MSFlexGrid1.TextMatrix(3, 4))
MSFlexGrid1.TextMatrix(5, 4) = Val(MSFlexGrid1.TextMatrix(5, 1)) *
Val(MSFlexGrid1.TextMatrix(5, 2))
MSFlexGrid1.TextMatrix(6, 4) = (Val(MSFlexGrid1.TextMatrix(6, 1)) *
Val(MSFlexGrid1.TextMatrix(6, 2))) + Val(MSFlexGrid1.TextMatrix(6, 4))
MSFlexGrid1.TextMatrix(7, 4) = (Val(MSFlexGrid1.TextMatrix(7, 1)) *
Val(MSFlexGrid1.TextMatrix(7, 2))) + Val(MSFlexGrid1.TextMatrix(7, 4))
MSFlexGrid1.TextMatrix(8, 4) = (Val(MSFlexGrid1.TextMatrix(8, 1)) *
Val(MSFlexGrid1.TextMatrix(8, 2))) + Val(MSFlexGrid1.TextMatrix(8, 4))
MSFlexGrid1.TextMatrix(9, 4) = (Val(MSFlexGrid1.TextMatrix(9, 1)) *
Val(MSFlexGrid1.TextMatrix(9, 2))) + Val(MSFlexGrid1.TextMatrix(9, 4))
MSFlexGrid1.TextMatrix(10, 4) = (Val(MSFlexGrid1.TextMatrix(10, 1)) *
Val(MSFlexGrid1.TextMatrix(10, 2))) + Val(MSFlexGrid1.TextMatrix(10, 4))
MSFlexGrid1.TextMatrix(11, 4) = Val(MSFlexGrid1.TextMatrix(11, 1)) *
Val(MSFlexGrid1.TextMatrix(11, 2))
MSFlexGrid1.TextMatrix(12, 4) = (Val(MSFlexGrid1.TextMatrix(12, 1)) *
Val(MSFlexGrid1.TextMatrix(12, 2))) + Val(MSFlexGrid1.TextMatrix(12, 4))
MSFlexGrid1.TextMatrix(13, 4) = (Val(MSFlexGrid1.TextMatrix(13, 1)) *
Val(MSFlexGrid1.TextMatrix(13, 2))) + Val(MSFlexGrid1.TextMatrix(13, 4))
MSFlexGrid1.TextMatrix(14, 4) = Val(MSFlexGrid1.TextMatrix(14, 1)) *
Val(MSFlexGrid1.TextMatrix(14, 2))
MSFlexGrid1.TextMatrix(15, 4) = (Val(MSFlexGrid1.TextMatrix(15, 1)) *
Val(MSFlexGrid1.TextMatrix(15, 2))) + Val(MSFlexGrid1.TextMatrix(14, 4))
MSFlexGrid1.TextMatrix(16, 4) = (Val(MSFlexGrid1.TextMatrix(16, 1)) *
Val(MSFlexGrid1.TextMatrix(16, 2))) + Val(MSFlexGrid1.TextMatrix(15, 4))
MSFlexGrid1.TextMatrix(17, 4) = Val(MSFlexGrid1.TextMatrix(17, 1)) *
Val(MSFlexGrid1.TextMatrix(17, 2))
MSFlexGrid2.ColWidth(0) = "1600"
MSFlexGrid2.ColWidth(1) = "525"
MSFlexGrid2.ColWidth(2) = "525"
MSFlexGrid2.ColWidth(3) = "525"
MSFlexGrid2.ColWidth(4) = "525"
MSFlexGrid2.ColWidth(5) = "525"
MSFlexGrid2.ColWidth(6) = "525"
MSFlexGrid2.ColWidth(7) = "525"
MSFlexGrid2.ColWidth(8) = "525"
MSFlexGrid2.ColWidth(9) = "525"
MSFlexGrid2.ColWidth(10) = "525"
MSFlexGrid2.ColWidth(11) = "525"
MSFlexGrid2.ColWidth(12) = "525"
MSFlexGrid2.ColWidth(13) = "700"
```

```
MSFlexGrid2.TextMatrix(1, 13) = toplam
MSFlexGrid2.TextMatrix(5, 0) = "Toplam Maliyet"
MSFlexGrid2.TextMatrix(0, 0) = "Aylar"
MSFlexGrid2.TextMatrix(1, 0) = "Net İhtiyaç"
MSFlexGrid2.TextMatrix(2, 0) = "Verilen Sipariş"
MSFlexGrid2.TextMatrix(3, 0) = "Stok Taşıma Maliyeti"
MSFlexGrid2.TextMatrix(4, 0) = "Hazırlık Maliyeti"
MSFlexGrid2.TextMatrix(0, 13) = "Toplam"
For y = 1 To 12
MSFlexGrid2.TextMatrix(0, y) = y
Next
MSFlexGrid2.TextMatrix(2, 1) = Val(MSFlexGrid2.TextMatrix(1, 1)) +
Val(MSFlexGrid2.TextMatrix(1, 2)) + Val(MSFlexGrid2.TextMatrix(1, 3))
MSFlexGrid2.TextMatrix(2, 4) = Val(MSFlexGrid2.TextMatrix(1, 4)) +
Val(MSFlexGrid2.TextMatrix(1, 5))
MSFlexGrid2.TextMatrix(2, 6) = Val(MSFlexGrid2.TextMatrix(1, 6)) +
Val(MSFlexGrid2.TextMatrix(1, 7))
MSFlexGrid2.TextMatrix(2, 8) = Val(MSFlexGrid2.TextMatrix(1, 8)) +
Val(MSFlexGrid2.TextMatrix(1, 9))
MSFlexGrid2.TextMatrix(2, 10) = Val(MSFlexGrid2.TextMatrix(1, 10)) +
Val(MSFlexGrid2.TextMatrix(1, 11))
MSFlexGrid2.TextMatrix(2, 12) = MSFlexGrid2.TextMatrix(1, 12)
For y = 1 To 11
If Val((MSFlexGrid2.TextMatrix(3, y))) >= Val((MSFlexGrid2.TextMatrix(1, y)))
Then
MSFlexGrid2.TextMatrix(3, y + 1) = Val(MSFlexGrid2.TextMatrix(3, y)) -
Val(MSFlexGrid2.TextMatrix(1, y))
Else
MSFlexGrid2.TextMatrix(3, y + 1) = Val(MSFlexGrid2.TextMatrix(2, y)) -
Val(MSFlexGrid2.TextMatrix(1, y))
End If
Next
For y = 1 To 12
If Val(MSFlexGrid2.TextMatrix(3, y)) = 0 Then
MSFlexGrid2.TextMatrix(4, y) = Val(S)
End If
Next
For j = 1 To 12
D = Val(MSFlexGrid2.TextMatrix(2, j))
top_sip = top_sip + D
MSFlexGrid2.TextMatrix(2, 13) = top_sip
Next
For j = 1 To 12
e = Val(MSFlexGrid2.TextMatrix(3, j))
top_stk_tsm = top_stk_tsm + e
MSFlexGrid2.TextMatrix(3, 13) = top_stk_tsm
Next
For j = 1 To 12
```

```
f = Val(MSFlexGrid2.TextMatrix(4, j))
top_hm = top_hm + f
MSFlexGrid2.TextMatrix(4, 13) = top_hm
Next
MSFlexGrid2.TextMatrix(5, 13) = Val(MSFlexGrid2.TextMatrix(3, 13)) +
Val(MSFlexGrid2.TextMatrix(4, 13))
End Sub
Private Sub Command9_Click()
MSFlexGrid2.Visible = True
MSFlexGrid1.ColWidth(0) = "2250"
MSFlexGrid2.ColWidth(0) = "2250"
MSFlexGrid1.Cols = 14
MSFlexGrid1.Rows = 6
MSFlexGrid2.Cols = 14
MSFlexGrid2.Rows = 6
MSFlexGrid1.ColWidth(0) = "1800"
MSFlexGrid1.ColWidth(1) = "500"
MSFlexGrid1.ColWidth(2) = "500"
MSFlexGrid1.ColWidth(3) = "500"
MSFlexGrid1.ColWidth(4) = "500"
MSFlexGrid1.ColWidth(5) = "500"
MSFlexGrid1.ColWidth(6) = "500"
MSFlexGrid1.ColWidth(7) = "500"
MSFlexGrid1.ColWidth(8) = "500"
MSFlexGrid1.ColWidth(9) = "500"
MSFlexGrid1.ColWidth(10) = "500"
MSFlexGrid1.ColWidth(11) = "500"
MSFlexGrid1.ColWidth(12) = "500"
MSFlexGrid1.ColWidth(13) = "650"
MSFlexGrid2.ColWidth(0) = "1800"
MSFlexGrid2.ColWidth(1) = "500"
MSFlexGrid2.ColWidth(2) = "500"
MSFlexGrid2.ColWidth(3) = "500"
MSFlexGrid2.ColWidth(4) = "500"
MSFlexGrid2.ColWidth(5) = "500"
MSFlexGrid2.ColWidth(6) = "500"
MSFlexGrid2.ColWidth(7) = "500"
MSFlexGrid2.ColWidth(8) = "500"
MSFlexGrid2.ColWidth(9) = "500"
MSFlexGrid2.ColWidth(10) = "500"
MSFlexGrid2.ColWidth(11) = "500"
MSFlexGrid2.ColWidth(12) = "500"
MSFlexGrid2.ColWidth(13) = "650"
MSFlexGrid2.ColAlignment(12) = 6
MSFlexGrid1.Width = "8500"
MSFlexGrid2.Width = "8500"
MSFlexGrid1.ColAlignment(13) = 6
MSFlexGrid2.ColAlignment(13) = 6
```

```
i = MSFlexGrid1.Row
j = MSFlexGrid1.Col
For j = 1 To 12
MSFlexGrid1.TextMatrix(0, j) = j
MSFlexGrid2.TextMatrix(0, j) = j
Next
hm = Val(InputBox("Hazırlık maliyetini giriniz."))
MSFlexGrid2.TextMatrix(5, 0) = "Toplam Maliyet"
MSFlexGrid1.TextMatrix(0, 0) = "Aylar"
MSFlexGrid1.TextMatrix(1, 0) = "Net İhtiyaç"
MSFlexGrid1.TextMatrix(2, 0) = "Parça-Dönem"
MSFlexGrid1.TextMatrix(4, 0) = "Parça-Dönem(kümülatif)"
MSFlexGrid1.TextMatrix(5, 0) = "Sipariş Miktarı"
MSFlexGrid1.TextMatrix(0, 13) = "Toplam"
MSFlexGrid2.TextMatrix(0, 0) = "Aylar"
MSFlexGrid2.TextMatrix(1, 0) = "Net İhtiyaç"
MSFlexGrid2.TextMatrix(2, 0) = "Verilen Sipariş"
MSFlexGrid2.TextMatrix(3, 0) = "Stok Taşıma Maliyeti"
MSFlexGrid2.TextMatrix(4, 0) = "Hazırlık Maliyeti"
MSFlexGrid2.TextMatrix(0, 13) = "Toplam"
For k = 1 To 12
b = Val(InputBox(k & ".ayın net ihtiyacını giriniz."))
MSFlexGrid1.TextMatrix(1, k) = b
MSFlexGrid2.TextMatrix(1, k) = b
toplam = toplam + b
Next
MSFlexGrid1.TextMatrix(1, 13) = toplam
MSFlexGrid1.TextMatrix(2, 1) = 0
MSFlexGrid1.TextMatrix(2, 2) = 1
MSFlexGrid1.TextMatrix(2, 3) = 2
MSFlexGrid1.TextMatrix(2, 4) = 3
MSFlexGrid1.TextMatrix(2, 5) = 1
MSFlexGrid1.TextMatrix(2, 6) = 2
MSFlexGrid1.TextMatrix(2, 7) = 1
MSFlexGrid1.TextMatrix(2, 8) = 2
MSFlexGrid1.TextMatrix(2, 9) = 1
MSFlexGrid1.TextMatrix(2, 10) = 2
MSFlexGrid1.TextMatrix(2, 11) = 1
MSFlexGrid1.TextMatrix(2, 12) = 2
For j = 2 To 12
MSFlexGrid1.TextMatrix(3, j) = MSFlexGrid1.TextMatrix(1, j)
Next
MSFlexGrid1.TextMatrix(4, 1) = 0
MSFlexGrid1.TextMatrix(4, 2) = MSFlexGrid1.TextMatrix(1, 2)
MSFlexGrid1.TextMatrix(4, 3) = Val(MSFlexGrid1.TextMatrix(4, 2)) +
(Val(MSFlexGrid1.TextMatrix(1, 3)) * Val(MSFlexGrid1.TextMatrix(2, 3)))
MSFlexGrid1.TextMatrix(4, 4) = Val(MSFlexGrid1.TextMatrix(4, 3)) +
(Val(MSFlexGrid1.TextMatrix(1, 4)) * Val(MSFlexGrid1.TextMatrix(2, 4)))
```

```
MSFlexGrid1.TextMatrix(4, 5) = MSFlexGrid1.TextMatrix(1, 5)
MSFlexGrid1.TextMatrix(4, 6) = Val(MSFlexGrid1.TextMatrix(4, 5)) +
(Val(MSFlexGrid1.TextMatrix(1, 6)) * Val(MSFlexGrid1.TextMatrix(2, 6)))
MSFlexGrid1.TextMatrix(4, 7) = MSFlexGrid1.TextMatrix(1, 5)
MSFlexGrid1.TextMatrix(4, 8) = Val(MSFlexGrid1.TextMatrix(4, 7)) +
(Val(MSFlexGrid1.TextMatrix(1, 8)) * Val(MSFlexGrid1.TextMatrix(2, 8)))
MSFlexGrid1.TextMatrix(4, 10) = Val(MSFlexGrid1.TextMatrix(4, 9)) +
(Val(MSFlexGrid1.TextMatrix(1, 10)) * Val(MSFlexGrid1.TextMatrix(2, 10)))
MSFlexGrid1.TextMatrix(4, 12) = Val(MSFlexGrid1.TextMatrix(4, 11)) +
(Val(MSFlexGrid1.TextMatrix(1, 12)) * Val(MSFlexGrid1.TextMatrix(2, 12)))
MSFlexGrid1.TextMatrix(4, 9) = MSFlexGrid1.TextMatrix(1, 5)
MSFlexGrid1.TextMatrix(4, 11) = MSFlexGrid1.TextMatrix(1, 5)
MSFlexGrid1.TextMatrix(5, 1) = Val(MSFlexGrid1.TextMatrix(1, 1)) +
Val(MSFlexGrid1.TextMatrix(1, 2)) + Val(MSFlexGrid1.TextMatrix(1, 3))
MSFlexGrid1.TextMatrix(5, 4) = Val(MSFlexGrid1.TextMatrix(1, 4)) +
Val(MSFlexGrid1.TextMatrix(1, 5))
MSFlexGrid1.TextMatrix(5, 6) = Val(MSFlexGrid1.TextMatrix(1, 6)) +
Val(MSFlexGrid1.TextMatrix(1, 7))
MSFlexGrid1.TextMatrix(5, 8) = Val(MSFlexGrid1.TextMatrix(1, 8)) +
Val(MSFlexGrid1.TextMatrix(1, 9))
MSFlexGrid1.TextMatrix(5, 10) = Val(MSFlexGrid1.TextMatrix(1, 10)) +
Val(MSFlexGrid1.TextMatrix(1, 11))
MSFlexGrid1.TextMatrix(5, 12) = MSFlexGrid1.TextMatrix(1, 12)
MSFlexGrid1.TextMatrix(5, 1) = Val(MSFlexGrid1.TextMatrix(1, 1)) +
Val(MSFlexGrid1.TextMatrix(1, 2)) + Val(MSFlexGrid1.TextMatrix(1, 3))
MSFlexGrid1.TextMatrix(5, 4) = Val(MSFlexGrid1.TextMatrix(1, 4)) +
Val(MSFlexGrid1.TextMatrix(1, 5))
MSFlexGrid1.TextMatrix(5, 6) = Val(MSFlexGrid1.TextMatrix(1, 6)) +
Val(MSFlexGrid1.TextMatrix(1, 7))
MSFlexGrid1.TextMatrix(5, 8) = Val(MSFlexGrid1.TextMatrix(1, 8)) +
Val(MSFlexGrid1.TextMatrix(1, 9))
MSFlexGrid1.TextMatrix(5, 10) = Val(MSFlexGrid1.TextMatrix(1, 10)) +
Val(MSFlexGrid1.TextMatrix(1, 11))
MSFlexGrid1.TextMatrix(5, 12) = MSFlexGrid1.TextMatrix(1, 12)
MSFlexGrid1.TextMatrix(5, 13) = MSFlexGrid1.TextMatrix(1, 13)
MSFlexGrid2.TextMatrix(2, 1) = Val(MSFlexGrid2.TextMatrix(1, 1)) +
Val(MSFlexGrid2.TextMatrix(1, 2)) + Val(MSFlexGrid2.TextMatrix(1, 3))
MSFlexGrid2.TextMatrix(2, 4) = Val(MSFlexGrid2.TextMatrix(1, 4)) +
Val(MSFlexGrid2.TextMatrix(1, 5))
MSFlexGrid2.TextMatrix(2, 6) = Val(MSFlexGrid2.TextMatrix(1, 6)) +
Val(MSFlexGrid2.TextMatrix(1, 7))
MSFlexGrid2.TextMatrix(2, 8) = Val(MSFlexGrid2.TextMatrix(1, 8)) +
Val(MSFlexGrid2.TextMatrix(1, 9))
MSFlexGrid2.TextMatrix(2, 10) = Val(MSFlexGrid2.TextMatrix(1, 10)) +
Val(MSFlexGrid2.TextMatrix(1, 11))
MSFlexGrid2.TextMatrix(2, 12) = MSFlexGrid2.TextMatrix(1, 12)
MSFlexGrid2.TextMatrix(3, 1) = 0
```

```
MSFlexGrid2.TextMatrix(3, 2) = Val(MSFlexGrid2.TextMatrix(1, 2)) +
Val(MSFlexGrid2.TextMatrix(1, 3))
MSFlexGrid2.TextMatrix(3, 3) = MSFlexGrid2.TextMatrix(1, 3)
MSFlexGrid2.TextMatrix(3, 4) = 0
MSFlexGrid2.TextMatrix(3, 5) = MSFlexGrid2.TextMatrix(1, 5)
MSFlexGrid2.TextMatrix(3, 6) = 0
MSFlexGrid2.TextMatrix(3, 7) = MSFlexGrid2.TextMatrix(1, 7)
MSFlexGrid2.TextMatrix(3, 8) = 0
MSFlexGrid2.TextMatrix(3, 9) = MSFlexGrid2.TextMatrix(1, 9)
MSFlexGrid2.TextMatrix(3, 10) = 0
MSFlexGrid2.TextMatrix(3, 11) = MSFlexGrid2.TextMatrix(1, 11)
MSFlexGrid2.TextMatrix(3, 12) = 0
For j = 1 To 12
If MSFlexGrid2.TextMatrix(3, j) = 0 Then
MSFlexGrid2.TextMatrix(4, j) = Val(hm)
End If
Next
MSFlexGrid2.TextMatrix(1, 13) = toplam
For j = 1 To 12
D = Val(MSFlexGrid2.TextMatrix(2, j))
top_sip = top_sip + D
MSFlexGrid2.TextMatrix(2, 13) = top_sip
Next
For j = 1 To 12
e = Val(MSFlexGrid2.TextMatrix(3, j))
top_stok_tasima = top_stok_tasima + e
MSFlexGrid2.TextMatrix(3, 13) = top_stok_tasima
Next
For j = 1 To 12
f = Val(MSFlexGrid2.TextMatrix(4, j))
top_hm = top_hm + f
MSFlexGrid2.TextMatrix(4, 13) = top_hm
Next
MSFlexGrid2.TextMatrix(5, 13) = Val(MSFlexGrid2.TextMatrix(3, 13)) +
Val(MSFlexGrid2.TextMatrix(4, 13))
End Sub
```