



Puan: 2,50

Soru 1

```
const char* AkilVer(){
    const char* akil = "AKIL";
    return akil;
}

int main(){
    printf("%s",AkilVer());
    return 0;
}
```

Şekildeki C kod parçası için ifade edilenlerden hangisi doğrudur?

- A** Ekrana null ifadesini yazar.
- B** Ekrana A harfini yazar.
- C** Ekrana AKIL yazar.
- D** Çalışma alanında hata verip sonlanır.
- E** Ekrana hiçbir şey yazmaz.

Seçimi Boş Bırakmak İstiyorum

Puan: 2,50

Soru 2

C# dilinde aşağıdaki parametre geçirme yöntemlerinden hangisi **yoktur**?

- A** Adres ile çağrıma
- B** Sonuç ile çağrıma
- C** Referans ile çağrıma
- D** Değer ile çağrıma
- E** İsim ile çağrıma

Seçimi Boş Bırakmak İstiyorum

Puan: 2,50

Soru 3

Uğurcan.yaz() – U,
}

Şekildeki Java kod parçasında parametre geçirme yöntemlerinden hangisi olduğu söylenebilir?

- A Referans ile çağrıma
- B Değer ile çağrıma
- C Adres ile çağrıma
- D Sonuç ile çağrıma
- E İsim ile çağrıma

Soru 4

Puan: 2,50

```
char* not="YT ";
void yaz(){
    printf("%s",not);
}
void FF(){
    not="FF ";
    yaz();
}
void DD(){
    char* not = "DD ";
    yaz();
    FF();
}
int main(){
    FF();
    DD();
    yaz();
    return 0;
}
```

Şekildeki kod parçasında **statik kapsam bağlama** kullanıldığı biliniyor. Ekran çıktısı ne olur?

- A FF FF FF FF
- B FF DD FF YT

44:57

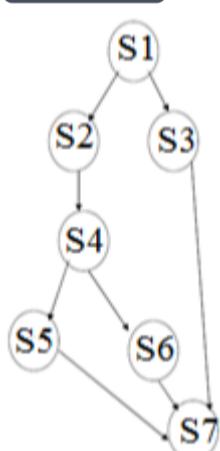
D YT DD FF FF**E** FF DD FF FF**Soru 5**

Puan: 2,50

C dili ile nesne yönelimli benzetim yapıldığında aşağıdakilerden hangisinin benzetiminin yapılması anlamsızdır?

A Sınıf**B** Arayüz**C** Kalıtım**D** Soyut Sınıf**E** Private Alan**Soru 6**

Puan: 2,50



Şekildeki öncelik grafına göre aşağıdaki deyimler eş zamanlı çalıştırılamaz?

A S3 – S4**B** S5 – S6**C** S3 – S6**D** S2 – S6**E** S2 – S3

```
(defun hadi(&optional A B C)
  (list A (list A B C) B C)
)
(hadi 6 3)
```

Şekildeki lisp kodunda Hadi fonksiyonu şekilde belirtildiği gibi çağrıldığında fonksiyonun döndürdüğü değer ne olur?

- A (6 (6 3 nil) 3 nil)
- B (6 3 nil)
- C nil
- D (6 (nil) 3)
- E (6 (6 3) 3)

Seçimi Boş Bırakmak İstiyorum

Soru 8

Puan: 2,50

Çoklu sınıf kalıtımına izin verilmeyen dillerde alternatif olarak kullanılan yöntem aşağıdakilerden hangisidir?

- A Soyut sınıflar
- B Namespace
- C Generic yapılar
- D Arayüzler
- E Object veri türü

Seçimi Boş Bırakmak İstiyorum

Soru 9

Puan: 2,50

C dilinde setjmp kütüphanesini ekleyen biri muhtemelen aşağıdaki işlemlerden hangisini yapacaktır?

- A Eşzamanlı Programlama
- B Dinamik Bölgede Bellek Ayırımı
- C Yapısal Programlama

E Nesne Yönelimli Benzetim

Soru 10

Puan: 2,50

```
i:=1;  
repeat  
    i:=i+1;  
until i<5;  
print(i);
```

Şekildeki kod parçası çalıştırıldığında ekrana yazacak olan i değerinin kaç olması beklenir?

- A** 4
- B** 2
- C** 3
- D** 5
- E** 1

Soru 11

Puan: 2,50

```
struct SAYI{  
    double deger;  
};  
typedef struct SAYI* Sayi;  
typedef struct SAYI Say;
```

Sayı s1;

```
int main(){  
    Say s2;  
    return 0;  
}
```

Şekildeki C kod parçası çalıştırıldığında hangi bellek bölgesinde herhangi bir değişken oluşmaz?



44:57

- A Heap Bellek Bölgesi
- B Ram belleği
- C Derlenmiş Kod Bölgesi
- D Çalışma Anı Yığını
- E Statik Bellek Bölgesi

Seçimi Boş Bırakmak İstiyorum**Soru 12**

Puan: 2,50

Bir prosesin bir işlemciye atanmak için bekletildiği durum aşağıdakilerden hangisidir?

- A New
- B Blocked
- C Running
- D Ready
- E Deadlock

Seçimi Boş Bırakmak İstiyorum**Soru 13**

Puan: 2,50

Java diline yeni eklenen **var** anahtar kelimesi için hangi tür bağlamaının olduğu söylenebilir?

- A Esnek Tip Bağlama
- B Dinamik Tip Bağlama
- C Statik Örtülü Tip Bağlama
- D Statik Dışsal Tip Bağlama
- E Yarı Dinamik Tip Bağlama

Seçimi Boş Bırakmak İstiyorum

Puan: 2,50



I- try
II- catch
III- finally
IV- throw
V- throws

A I - II - III

B I - II - III - IV

C I - II - IV - V

D I - II

E I - II - IV

Seçimi Boş Bırakmak İstiyorum

Puan: 2,50

Soru 15
(setq x #b010 y #b100)
(setq x (* x y))
(decf x)

Arka arkaya sırasıyla şekildeki gibi girilen Lisp komutlarından sonra x'in değeri kaç olur?

A 1000
B 3
C 999
D 7
E 9

Seçimi Boş Bırakmak İstiyorum

Puan: 2,50

Soru 16

Aşağıda verilen programlama dilleri değerlendirme kriterlerinden hangi ikilisinde biri yükselirken diğerinde düşüş beklenir?

A Nesne Yönelimlilik – Modülerlik



44:57

C Okunabilirlik – Öğrenme Kolaylığı

D Esneklik - Güvenlik

E Taşınabilirlik – Verimlilik

Puan: 2,50

Soru 17

Mantıksal programlamada kısıtlama portlarında aşağıdakilerden hangisi yoktur?

A fail

B call

C undo

D exit

E redo

Puan: 2,50

Soru 18

Ogrenci o1=null;

if(o1 != null && o1.isim == "Mehmet") print("Merhaba");

if(o1 == null) print("Güle güle");

Kısa devre değerlendirmenin olduğu bir dilde şekildeki kod için hangisi söylenebilir?

A Çalışma anında hata verir.

B Derlenme anında hata verir.

C Ekrana Güle güle yazar.

D Ekrana hem Merhaba hem de Güle güle yazar.

E Ekrana Merhaba yazar.

Puan: 2,50

44:57

S → ε | εT

S → ε

Şekilde CFG ile ifade edilmiş dilbilgisinden, aşağıdaki karakter katarlarından hangisi üretilemez?

- A 11
- B ε
- C 0
- D 1
- E 101

Soru 20

Puan: 2,50

```
char* AkilVer(){  
    char akil[] = "AKIL";  
    return akil;  
}  
  
int main(){  
    printf("%s",AkilVer());  
    return 0;  
}
```

Şekildeki C kod parçası için ifade edilenlerden hangisi doğrudur?

- A Ekrana A harfini yazar.
- B Ekrana null ifadesini yazar.
- C Ekrana hiçbir şey yazmaz.
- D Çalışma anında hata verip sonlanır.
- E Ekrana AKIL yazar.

Soru 21

Puan: 2,50



44:57

Şekilde verilmiş BNF'yi doğru ifade eden EBNF hangi şıkta verilmiştir?

- A $\langle id \rangle ::= (\langle karakter \rangle \mid \langle rakam \rangle)^*$
- B $\langle id \rangle ::= (\langle karakter \rangle \mid \langle karakter \rangle \langle rakam \rangle)^+$
- C $\langle id \rangle ::= \langle karakter \rangle (\langle karakter \rangle \mid \langle rakam \rangle)^*$
- D $\langle id \rangle ::= (\langle karakter \rangle \mid \langle rakam \rangle)^+$
- E $\langle id \rangle ::= \langle karakter \rangle \mid (\langle karakter \rangle \mid \langle rakam \rangle)^+$

Soru 22

Puan: 2,50

C dilinde tanımlanmış tek boyutlu bir tamsayı dizisinin ilk eleman adresi 0x00FA1C olduğu biliniyor her bir hücre 4 byte yer kaplıyorsa 8. elemanın bellekteki adresi aşağıdakilerden hangisidir?

- A 0x00FA28
- B 0x00FA30
- C 0x00FA3C
- D 0x00FA38
- E 0x00FA34

Soru 23

Puan: 2,50

Aşağıdaki programlama dillerinden hangisi makro diller grubunda bulunmaz?

- A Python
- B Excel
- C Perl



44:57

E Javascript**Soru 24**

Puan: 2,50

```
struct SAYI{  
    double deger;  
};  
typedef struct SAYI* Sayi;  
typedef struct SAYI Say;
```

```
Sayi s1;
```

```
int main(){  
    Say s2;  
    return 0;  
}
```

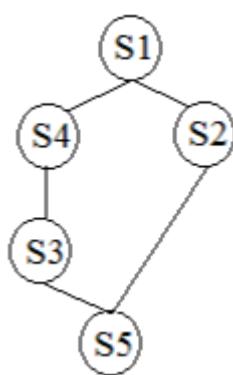
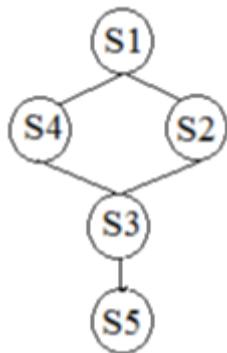
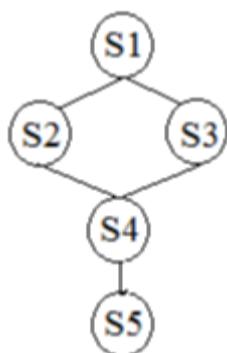
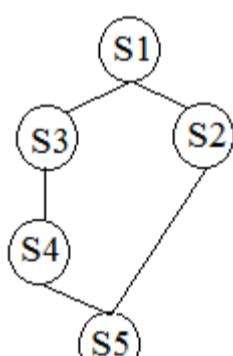
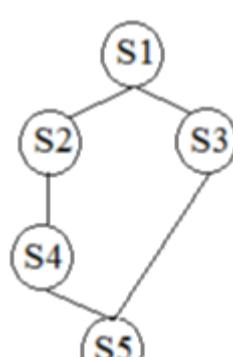
Şekildeki C kod parçası için ifade edilenlerden hangisi yönlüktür?

- A** s2 içinde deger isimli değişkeni barındırır.
- B** Sayı arkasında bir adres barındırır.
- C** s1 sadece bir referanstır.
- D** s1 ve s2'nin bellekte kapladıkları alan eşittir.
- E** Say arkasında yapının kendisini barındırır.

Soru 25

Puan: 2,50

İşte sorular
sorular.

A**B****C****D****E**

Seçimi Boş Bırakmak İstiyorum

Soru 26

Puan: 2,50

Aşağıdakilerden hangisi nesnenin adresi yerine nesnenin kendisini içerisinde barındırır?



44:57

- B** Ogrenci o1; (C++)
- C** Ogrenci o1; (C#)
- D** o1 = Ogrenci() (Python)
- E** Hiçbiri

Seçimi Boş Bırakmak İstiyorum

Puan: 2,50

Soru 27

Derleyici ve yorumlayıcı ile ilgili aşağıdaki ifadelerden hangisi yanlıştır?

- A** Yorumlayıcıda tespit edilen hatalar kullanıcıya direkt bildirilir.
- B** Yorumlayıcı derleyiciye göre daha çok bellek kullanır.
- C** Yorumlayıcıda bir satırın çalışması için gereken çevrim süreci her çalışmada tekrarlanır.
- D** Yorumlayıcıda dil dönüşümü satır satır gerçekleşir.
- E** Derleyici kaynak kodu bütün olarak makine diline çevirir.

Seçimi Boş Bırakmak İstiyorum

Puan: 2,50

Soru 28

```
int a=1;  
printf("%d %d %d %d",++a,a++,a,++a);
```

Şekildeki C kod parçası Mingw'de derlenip çalıştırıldığında ekrana ne yazacaktır?

- A** 4 2 4 4
- B** 2 2 3 4
- C** 4 4 2 4
- D** 3 2 2 2
- E** 2 2 2 3

Seçimi Boş Bırakmak İstiyorum

Puan: 2,50

Soru 29

- A $S \rightarrow 1S \mid 13 \mid \epsilon$
- B $S \rightarrow S3 \mid S1 \mid \epsilon$
- C $S \rightarrow S3 \mid 1S \mid \epsilon$
- D $S \rightarrow 1S \mid \epsilon$
- E $S \rightarrow 3S \mid 1S \mid \epsilon$

Soru 30

Puan: 2,50

```
char* not="YT ";
void yaz(){
    printf("%s",not);
}
void FF(){
    not="FF ";
    yaz();
}
void DD(){
    char* not = "DD ";
    yaz();
    FF();
}
int main(){
    FF();
    DD();
    yaz();
    return 0;
}
```

Şekildeki kod parçasında **dinamik kapsam bağlama** kullanıldığı biliniyor. Ekran çıktısı ne olur?

- A FF DD FF YT
- B YT YT YT YT



44:57

 D FF DD FF FF E FF FF FF FF

Seçimi Boş Bırakmak İstiyorum

Soru 31

Puan: 2,50

Programlama dilleri tasarım paradigmalarına göre sınıflandırıldığında aşağıdakilerden hangisi bu listede **yer almaz**?

 A Nesne Yönelimli Programlama B Yapısal Programlama C Fonksiyonel Programlama D Emir Esaslı Programlama E Mantıksal Programlama

Seçimi Boş Bırakmak İstiyorum

Soru 32

Puan: 2,50

S0

Parbegin

Begin

S1

S2

End

S3

Begin

S4

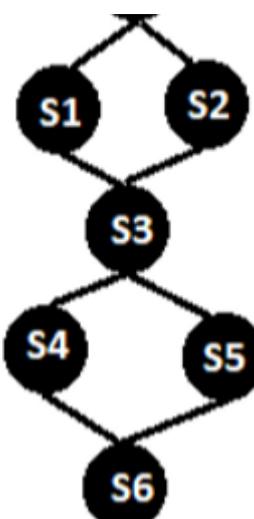
S5

End

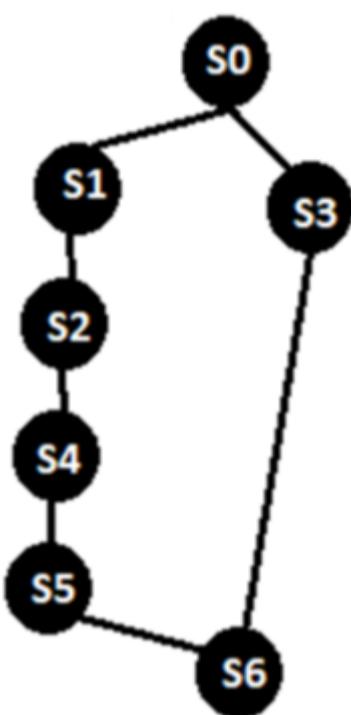
ParEnd

S6

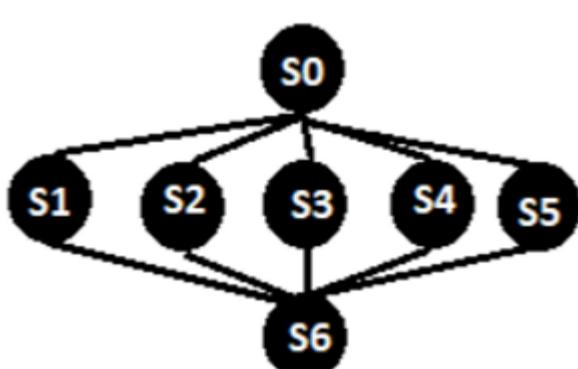
Şekildeki parbegin ve parend yapısının öncelik grafi aşağıdakilerden hangisidir?



B

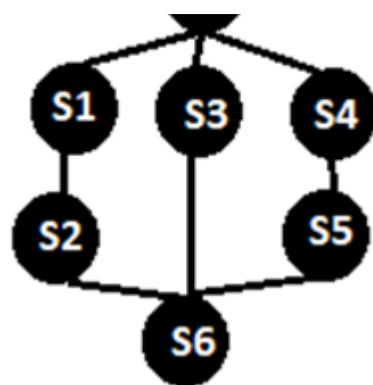


C



D





Soru 33

Puan: 2,50

Kurucu metotlarla ile ilgili aşağıdakilerden hangisi ya da hangileri yanlıştır?

- I- Yalnız parametresiz kurucu yazılabilir.
- II-Kurucular sınıfından farklı bir isme sahip olmalıdır.
- III-Bir nesne oluşturulurken çağrılan metottur.
- IV-Yapılandırıcılar adres döndürür.

- A II - III - IV
- B Hepsi
- C I - II
- D I - IV
- E I - II - IV

Soru 34

Puan: 2,50

```
int a=1;  
System.out.printf("%d %d %d %d",++a,a++,a,++a);
```

Şekildeki Java kod parçası NetBeans'te derlenip çalıştırıldığında ekrana ne yazacaktır?

- A 4 4 2 4
- B 2 2 3 4
- C 3 2 2 2



44:57

E 4 2 4 4**Seçimi Boş Bırakmak İstiyorum****Soru 35**

Puan: 2,50

Bazı programlama dilleri her iki tasarım paradigmalarını içinde barındırır. Aşağıdakilerden hangisi buna örnektir?

A C#**B** C**C** Prolog**D** C++**E** Java**Seçimi Boş Bırakmak İstiyorum****Soru 36**

Puan: 2,50

Aşağıdakilerden hangisi derlenme sonucu ortaya çıkan ürünün uygulama alanlarına göre gruplandırıldığından bu grplardan biri değildir?

A Bilimsel ve Mühendislik Yazılımları**B** Görüntüsel Yazılımlar**C** Yapay Zeka Yazılımları**D** Yüksek Seviyeli Yazılımlar**E** Mesleki Yazılımlar**Seçimi Boş Bırakmak İstiyorum****Soru 37**

Puan: 2,50

<reel> ::= <tam>.<kesir> | .<kesir>

Şekilde verilmiş olan BNF'yi doğru ifade eden EBNF hangi şıkta doğru verilmiştir?

A **<reel> ::= <tam>[.<kesir>]**



44:57

- C <reel> ::= [<tam>.<kesir>]
- D <reel> ::= [<tam>].<kesir>
- E <reel> ::= <tam>[.]<kesir>

Seçimi Boş Bırakmak İstiyorum**Soru 38**

Puan: 2,50

```
Ogrenci o1=null;  
if(o1 != null && o1.isim == "Mehmet") print("Merhaba");  
if(o1 == null) print("Güle güle");
```

Kısa devre değerlendirmenin olmadığı bir dilde şekildeki kod için hangisi söylenebilir?

- A Derlenme anında hata verir.
- B Ekrana Güle güle yazar.
- C Çalışma anında hata verir.
- D Ekrana Merhaba yazar.
- E Ekrana hem Merhaba hem de Güle güle yazar.

Seçimi Boş Bırakmak İstiyorum**Soru 39**

Puan: 2,50

```
int kontrol=0;  
double a=10,b=3,c;  
float x=10,y=3,z;  
c=a/b;  
z=x/y;  
if(sizeof(a) > sizeof(b)) kontrol++;  
if(sizeof(x) > sizeof(y)) kontrol++;  
if(sizeof(a) > sizeof(x)) kontrol++;  
if(c == z) kontrol++;
```

Şekildeki kod parçasında kontrol değişkeninin son değeri kaç olur?



44:57

B 4**C** 1**D** 3**E** 2 Seçimi Boş Bırakmak İstiyorum**Soru 40**

Puan: 2,50

Aşağıdakilerden hangisi genel amaçlı programlama dili olarak **gösterilemez**?

A Cobol**B** Fortran**C** Basic**D** Java**E** C Seçimi Boş Bırakmak İstiyorum Cevapları Gözden Geçir

PROGRAMLAMA DİLLERİNİN GELİŞİMİ

- Programlama dili tasarım ve gerçekleştirimleri, 1950'li yıllarda tanıtılan ilk yüksek düzeyli diller olan FORTRAN, COBOL ve LISP'den beri sürekli olarak gelişmiştir.
- Günümüzde hızla değişen bilgisayar teknolojileri, yeni gereksinimleri ortaya çıkarmaktadır. Bunun sonucu olarak, gelecekte de yeni programlama dillerinin geliştirilmesi kaçınılmazdır.

Programlama Dillerinin Sınıflandırılması (Seviyesine göre)

- Çok Yüksek Seviyeli Programlama Dilleri (insana en yakın) VISUAL BASIC, Access....(Dekleratif Diller)
- Yüksek Seviyeli Programlama Dilleri (PASCAL, COBOL)
- Orta Seviyeli Programlama Dilleri (C, ADA)
- Alçak Seviyeli Programlama Dilleri (Sembolik Makine Dilleri)
- Makine Dilleri (Bilgisayara en yakın)

Dillerdeki seviye yükseldikçe programcının işi daha kolay hale gelirken genel olarak esneklik ve verimlilik azalmaktadır.

Uygulama Alanlarına Göre

- Sayısal Uygulamalar için programlama dilleri
- Ticari Uygulamalar için programlama dilleri
- Yapay Zeka Uygulamaları için programlama dilleri
- Sistem programlama için programlama dilleri

Sayısal Uygulamalara Yönelik Programlama Dilleri

- Bilgisayarların ilk olarak kullanıldıkları alan sayısal uygulamaların ağırlıklı olduğu bilimsel çalışmalar olmuştur. Bu nedenle ilk geliştirilen programlama dilleri, sayısal programlama özelliklerini vurgulamışlardır.

FORTRAN

ALGOL

PL/I

BASIC

APL

SIMULA 67

PASCAL

C

ADA

Ticari Uygulamalara Yönelik Programlama Dilleri

- Ticari uygulamalardaki veri işleme, sayısal hesaplamalardan sonra ilk olarak gelişen uygulama alanıdır.
- **COBOL** (CCommon Bussiness Oriented Language) :
- A.B.D. Savunma Bakanlığı'nın, birçok şirkete birlikte İngilizce'ye yakın ve ticari uygulamalara yönelik bir dilin geliştirilmesi çalışmalarını desteklemesi sonucu, 1959 yılında COBOL tanıtılmıştır.
- COBOL dili, özellikle yoğun miktarda veri işleme kolaylıklarını sağlayan deyimleri ve yapıları nedeniyle ticari uygulamalar alanında yazılım geliştirmek için popüler olmuştur.
- COBOL, hiyerarşik veri yapıları gibi birçok yeni kavram içeren ve özellikle raporlama açısından çeşitli olanaklar sağlayan bir dildir. 1961 ve 1962'de yenilenen dil, 1968'de standartlaştırılmış ve 1984'de tekrar yenilenmiştir.

Yapay Zeka Uygulamaları İçin Programlama Dilleri

- **LISP :** LISP 1950'li yılların sonunda, *liste işleme* amaçlı fonksiyonel bir dil olarak, John McCarthy tarafından IBM 704 bilgisayarları için geliştirilmiştir. LISP, diferansiyel ve integral hesaplamalarında, sayısal mantık ve yapay zekanın diğer alanlarında sembolik hesaplamalar için kullanılmıştır. Sonraki yıllarda, birçok kez yenilenen LISP'ten başka, Scheme (1975) ve ML (1988) de LISP'i izleyen yapay zeka alanındaki fonksiyonel dillere örnektir.
- **PROLOG:** Prolog, temel denetim yapısı sembolik mantık kavramlarına dayanan özel amaçlı bir dildir. 1972 yılında tanıtılmış olan Prolog'un temel uygulama alanı, doğal dil işlemedir. Günümüze kadar Prolog, veritabanlarından uzman sistemlere kadar çeşitli uygulama alanlarında kullanılmıştır.

GİRİŞ/ÇIKIŞ KOLAYLIĞI

- Sıralı, indexli ve rastgele dosyalara erişme, veritabanı kayıtlarını geri alma, güncelleştirme ve sorgulama yeteneği olarak tanımlanabilir.
- C dili bu bakımdan zayıftır.
- Genel olarak veritabanı programları bu bakımdan güçlündür.

VERİMLİLİK

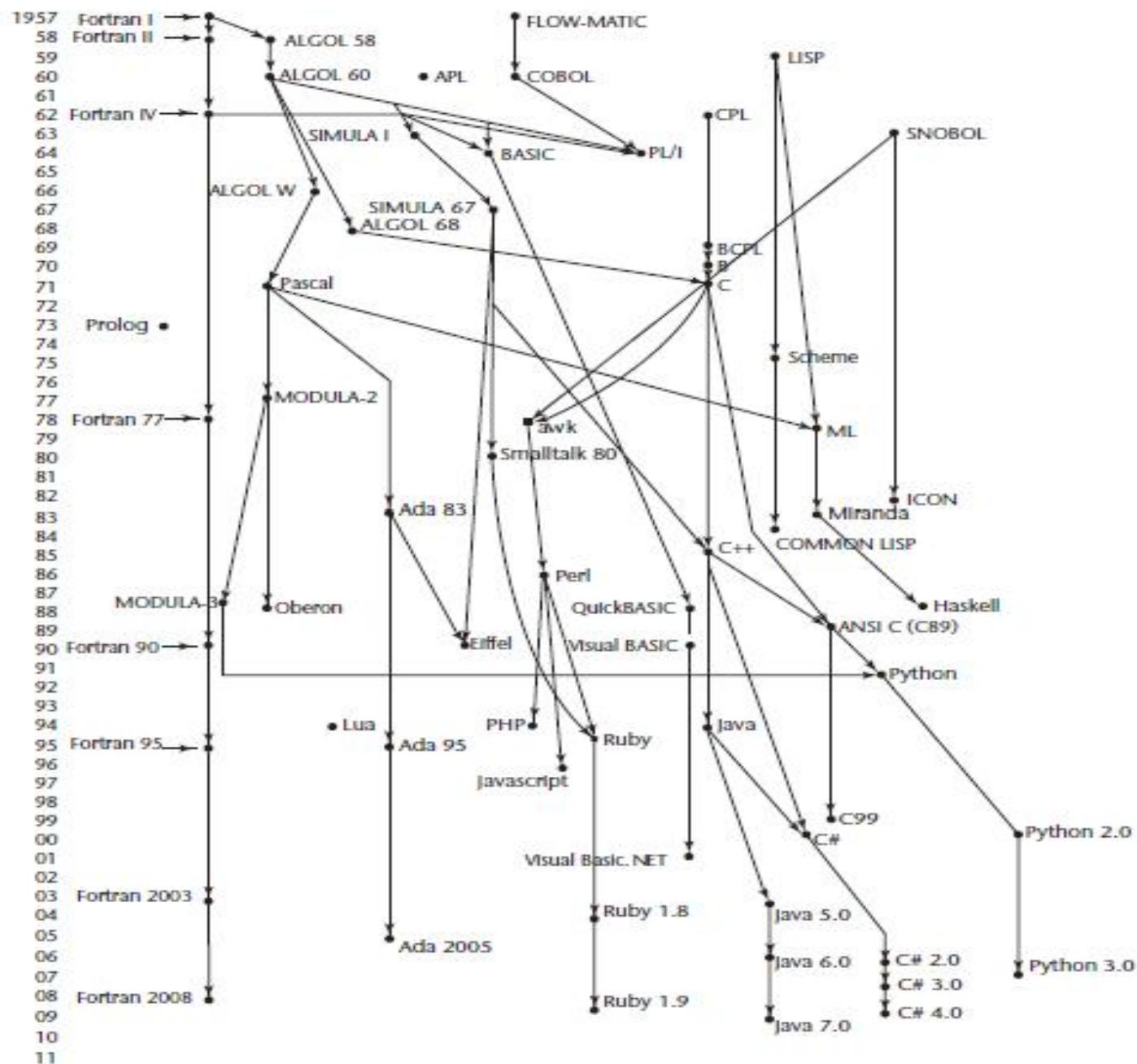
- Bir dilde yazıldıktan sonra amaç koda dönüştürülmüş programların hızlı çalışılmasına verimlilik denir.
- C programları hızlı çalışır ve az yer kaplar.
- Çalışabilir kodun küçüklüğü ile çalışma hızı arasında doğrusal bir ilişki vardır.

YAPISALLIK

Burada bloklar halinde yazım ön plandadır.

- Yoğun olarak altprogram kullanılmaktadır.
- Program akışında atlamaların yapılması okumayı ve algılamayı zorlaştırabilir.
- Altprogramlar sayesinde soyutlama söz konusudur.

Evolution of the Major Programming Languages



Fortran (devam)

- Fortran ingilizce sözcüklerden oluşuyordu
- Anlaşılması makine diline göre çok kolaydı
- Matematiksel uygulamalar için iyi özelliklere sahipti.
- Fakat;
- Fortran dilinin bilgisayar tarafından anlaşılması için DERLEYİCİ'ye ihtiyacı vardı.
- John Backus, 2 yılda, 51000 satırlık makine kodundan oluşan ve bir teyp biriminde saklanan Fortran derleyicisini üretmiştir.
- FORTRAN 1957 yılında ticari olarak kullanıma sunuldu
- Aynı kişi 1959 yılında, yüksek seviyeli bir dilin yazış kurallarını açıklama noktasında standart bir notasyon haline gelen **Backus Naur Form (BNF)**'yi bulmuştur.

FORTRAN ailesi

- FORTRAN I:Taşınabilirliği çok kötü
- FORTRAN 66:Karakter türü verileri işlemeye çok kısıtlı, yapısal programlama yok, reküratif işlemler yapılamıyor.
- FORTRAN 77(American National Standards Institute-ANSI): Karakter türü verileri işleyebiliyor, Yapısal programlamayı destekliyor, DO çevrimine dışarıdan veri girilebiliyor. Taşınabilirlik daha iyi hale getiriliyor.
- FORTRAN 90:Pointer (bağlı liste gerçeklemesi, dinamik bellek yönetimi), reküratif işlem yeteneği, bit düzeyinde işlem, Dizi yapıları daha iyi kullanılabiliyor, Altprogram yapıları daha esnek, isimler daha uzun yazılabiliyor (okunabilirlik)

LISP (devam)

- LISP esas olarak yorumlayıcı kullanan bir dildir. Derleyici kullanan versiyonları da vardır.
- Veri tiplemesi bakımından esnek bir dildir.
- Olağanüstü esneklik, ifade gücü ve **kodun aynı zamanda veri olarak da kullanılabilmesi** özelliği LISP'i yapay zaka uygulamalarında rakipsiz hale getirmiştir.
- Nesne Yönelimli Programlamayı destekler
- Veri tabanlarına erişim ve GUI olanağı vardır.
- Çok iyi belgelendirilmiş olup, COMMON LISP, ANSI tarafından standart hale getirilmiştir.

ALGOL

- ALGOrithmic Language
- FORTRAN I'den esinlenilmiştir
- İlk kez 1958 yılında Avrupalı ve Amerikalı bir komisyonun Zürih'teki çalışmaları sonucu oluşturulan yüksek seviyeli bir dildir. İlk isim ALGOL 58 olarak verilmiştir.

ALGOL 60

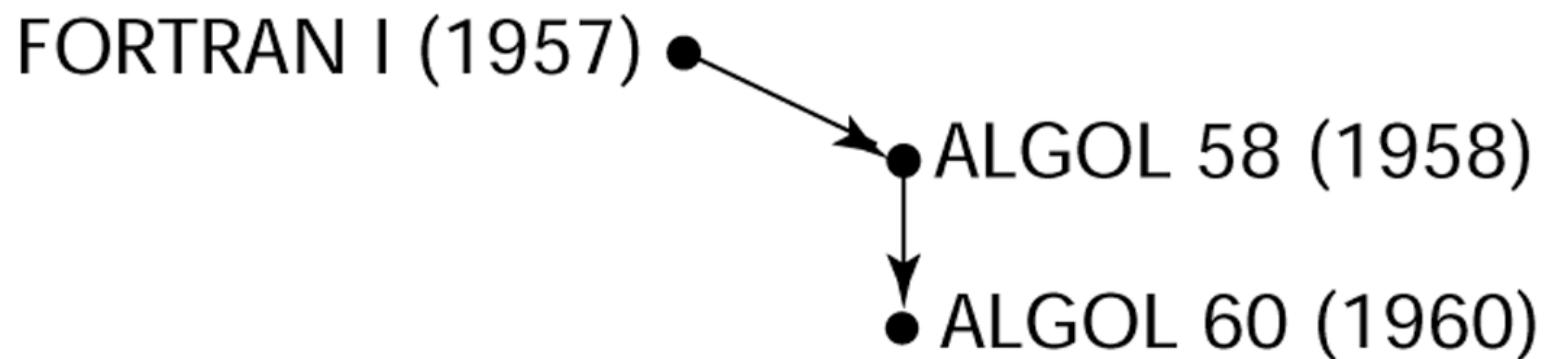
- GAMM(German Society for Applied Mathematics and Mechanics) tarafından ALGOL'58'e eklemeler yapılmıştır.
- Amaçlar:
 - Matematiksel notasyonlara yakın ve kolay okunabilen
 - Literatürdeki hesap süreçlerinde kullanılabilen
 - Makine diline kolaylıkla çevrilebilen bir dil olsun
- Makineden bağımsız, daha esnek ve daha güçlüğün olmustur.
- Atama ifadesi olarak ilk evrensel kullanım bu dilde olmuştur.

variable := expression

ALGOL 60: Eksik yönler

- Anlaşılmakta zorluk ve yürütmede (implementation) verimsizlige götürecek kadar esnek
- I/O ifadelerindeki yetersizlik yada zayıflıklar (eksiklik)
- 1960'lı yıllar için BNF(Backus Naur Form) kullanımı karmaşıklıklaşmış gibi gözükmeğtedir.
- Avrupalı bilim adamları arasında sonderece popüler, eğitim ve araştırma aracı olarak kullanılır ve bilimsel makalelerde algoritmaları açıklamak in kullanılmasına rağmen;
- IBM tarafından desteklenmemiştir.
- Çünkü bu sırada IBM FORTRAN dilinde yazılmış zengin bir kütüphaneye sahiptir ve haliyle FORTRAN' desteklemektedir.
- O zamanlar IBM'in bilişim sektörünün %80'ine sahip olduğu düşünülürse, bu durum ALGOL60 için bir dezavantajdır.

ALGOL 60'ın şeceresi

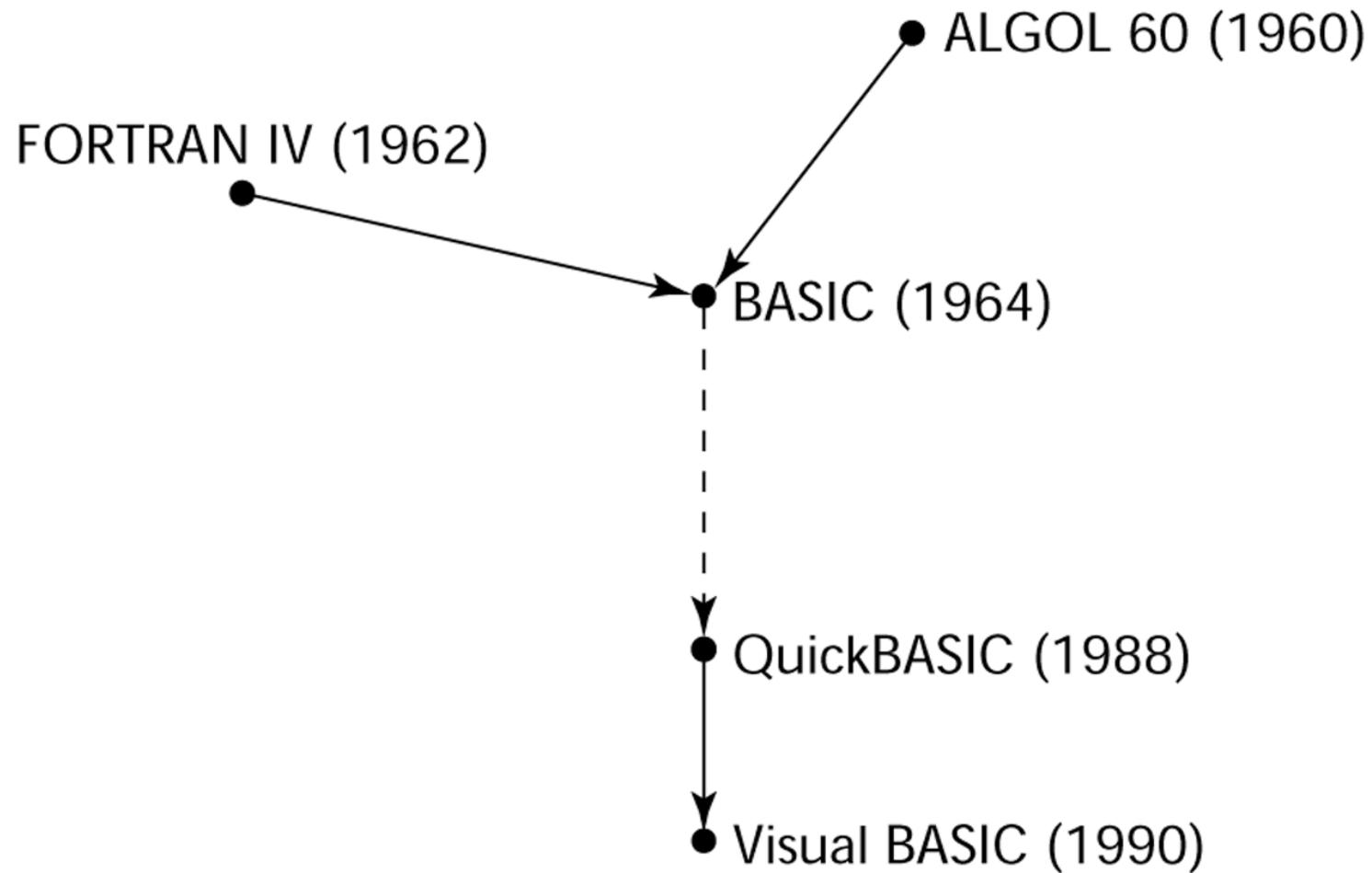


BASIC: İlk Zaman Paylaşımı Bilgisayar Sistemi (1964)

- Beginner's All-purpose Symbolic Instruction Code
- Öğrencilerin bilgisayara daha kolay erişimlerini sağlamak ve basit ve etkin bir programlama dili ile program yazabilme isteklerine cevap vermek için tasarlanmış bir dildir.
- The first PL used through terminals connecting to a remote computer
- Sadece 14 komuta (LET, PRINT, GOTO...) sahipti. Tek veri tipi (number= kayan noktalı ve tamsayı)
- BASIC, FORTRAN ve ALGOL'den bazı bileşenleri almıştır. FORTRAN'dan DO çevrimini, ALGOL'den ise "until" yerine "to"

- Kolay bir dil ve genel maksatlı, belirli bir alana bağlı değil
- Uzman kişilere de hitap edebiliyor
- Açık ve anlaşılır hata mesajlarına sahip, kullanıcı bilgisayarla etkileşimli çalışabiliyor
- Küçük boyutlu programları hızlı bir biçimde çalıştırabiliyor
- Kullanım için donanım bilgisine sahip olmaya gerek yok
- Kullanıcıyı işletim sistemi ayrıntılarından dahi koruyabiliyor
- Derleyici kullanıyor, programın tümü makine diline çevrildikten sonra icra ediliyor
- BASIC'in pekçok versiyonları olmuştur. 1989'da ise nesne yönelimli uyarlama olan Visual BASIC ve 1998'de VB6.0 sunulmuştur.

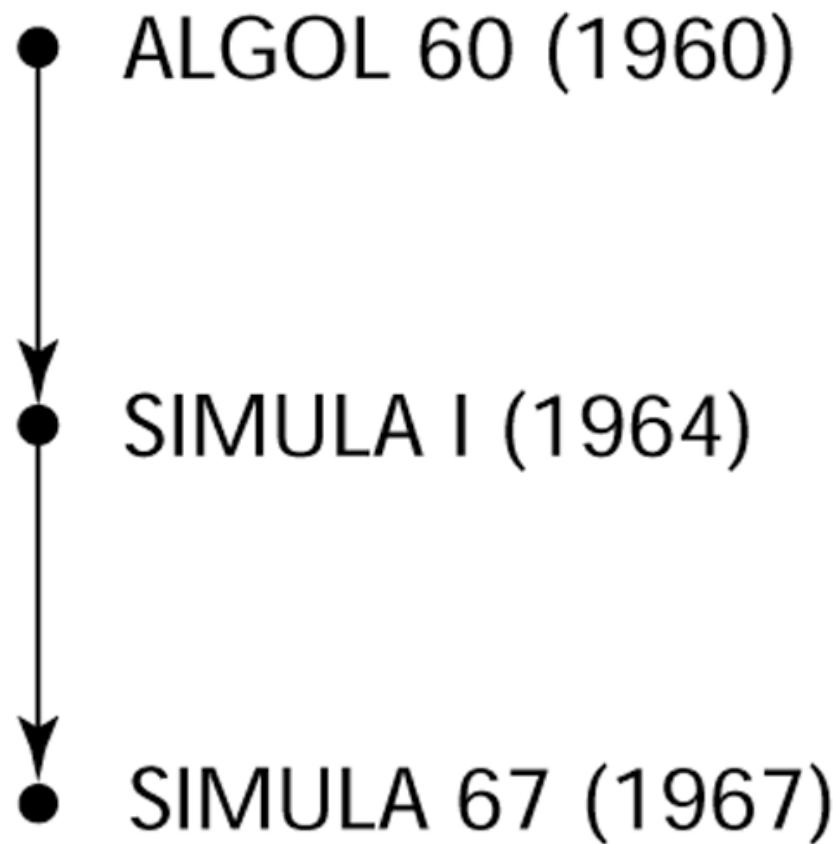
BASIC'in Şeceresi



SIMULA 67: Beginnings of Data Abstraction (1962 ve 1964 arasında)

- İlk olarak simülasyon için tasarlanmış bir dildir.
- ALGOL60'ın genişletilmiş versiyonudur. (Blok yapısı ve kontrol ifadeleri buradan alınmıştır)
- Daha önce durdurulduğu yerden itibaren yeniden çalışmaya başlayan altprogramları desteklemektedir.
- Veri soyutlamasına imkan veren sınıf yapılarını desteklemektedir

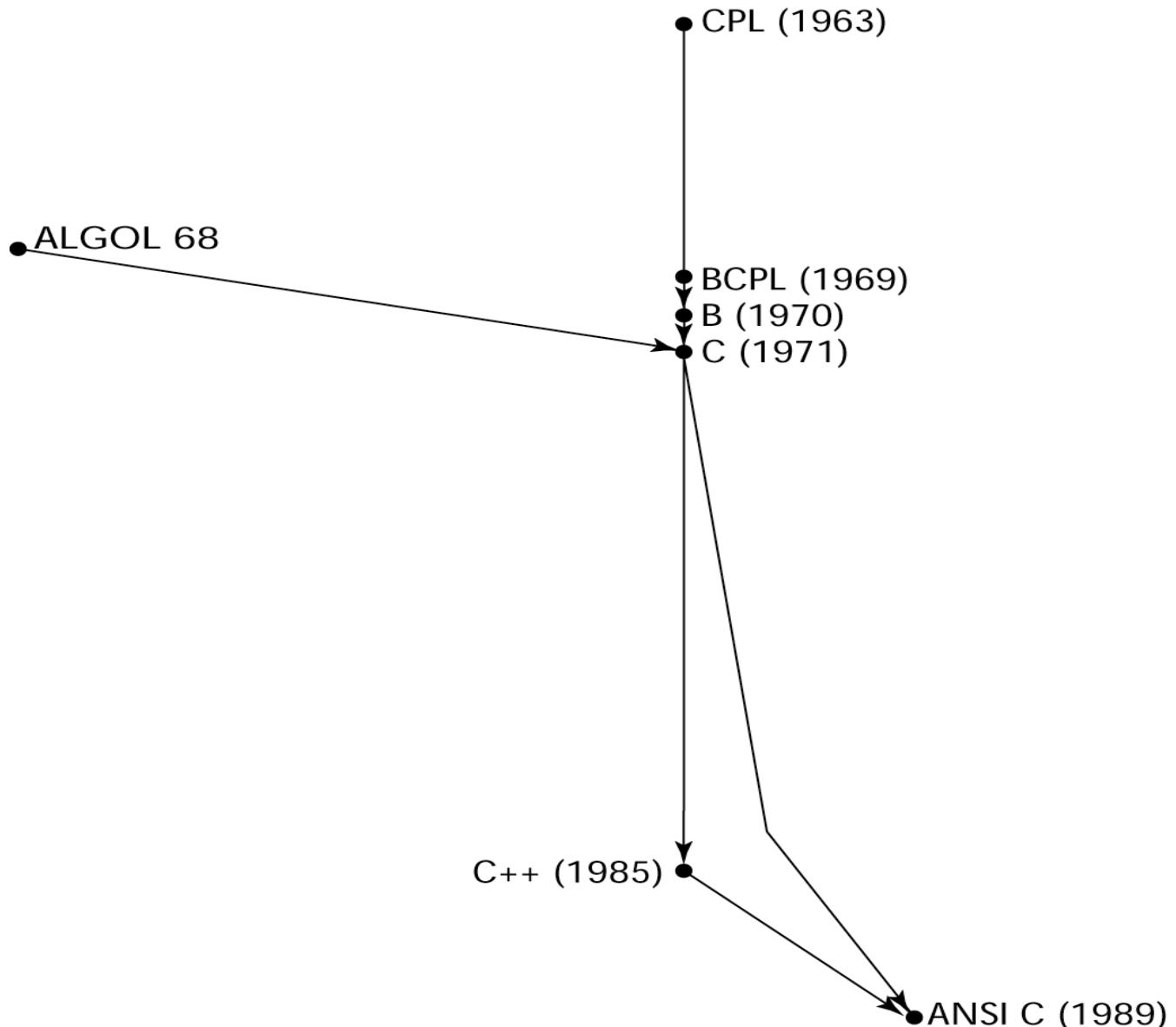
SIMULA 67'in Şeceresi



ALGOL'ün Torunları

- **Pascal (1971):**
 - İlk önceleri eğitim dili olarak kullanıldı
 - Basit ve okunabilir, yazılabilir (expressive)
 - Fortran and C ile karşılaştırıldığında emniyetli bir dildir.
- **C: Taşınabilir Sistem Programlama Dili (1972)**
 - Yeterli derecede kontrol ifadelerine ve veri yapısı olanaklarına sahip.
 - Tip kontrolü yetersiz.
 - Derleyicilerine çok kolay bir biçimde ulaşılabilir.

C'nin Şeceresi



ADA:

Tarihin en geniş tasarım çalışması

- ABD savunma bakanlığının bir çalışması sonucu ortaya çıkmıştır. Gömülü sistemlerin (embedded systems) programlanması sağlayacak PL üretimi amaçlanmıştır. (1983). Gerçek zamanlı uygulamlar için
- Augusta Ada Byron'ın (1815-1851) ölümünden sonra isimlendirilmiştir.
- Blok yapılı, nsne yönelimli, genel amaçlı ve eşzamanlılığı destekleyen bir dildir.
- Büyük boyutlu yazılımlar için uyundur.

ADA (devam)

- Şablon (Generic) program birimleri sayesinde yazılımın yeniden kullanılabilmesini Buluşma yeri(rendezvous) mekanizmasının ilavesiyle eşzamanlı çalışmayı desteklemektedir. mechanism
- Çok geniş ve çok karmaşık bir dil. (Özellikle yazım kuralları)
- Çeşitli durumlara uygulanabilecek hazır şablonlara (template) sahiptir.
- İlk sıralar ADA derleyicileri kod üretmekte verimsiz idi.
- En çok gömülü sistemlerde başarılı olmuştur.
- Veri tipleri konusunda çok zengindir. Çok-iş işleme özelliğine sahiptir.

Ada 95

- ADA83'e kalıtım (**inheritance**), çok biçimlilik (**polymorphism**) gibi yeni eklemeler yapılarak ve **tip türetim mekanizmasını genişleterek** Ada 95 elde edilmiştir.
- Altprogramların dinamik kapsam bağlama kurallarına göre çağrılmaması mekanizması da ADA 95'in özelliklerine katılmıştır.

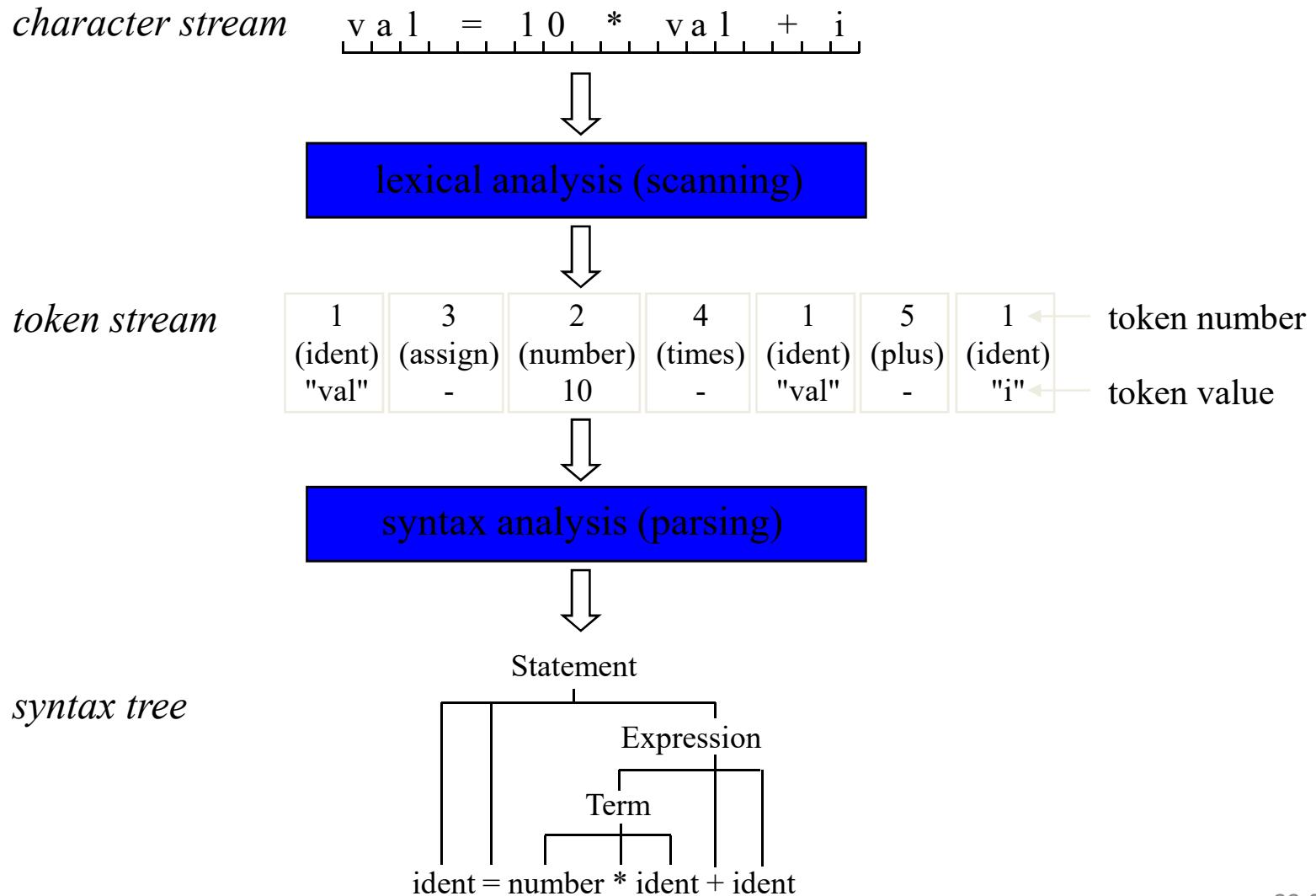
Ada 95

- ADA83'e kalıtım (inheritance), çok biçimlilik (polymorphism) gibi yeni eklemeler yapılarak ve tip türetim mekanizmasını genişleterek Ada 95 elde edilmiştir.
- Altprogramların dinamik kapsam bağlama kurallarına göre çağrılmaması mekanizması da ADA 95'in özelliklerine katılmıştır.

ADA'nın şeceresi

- Pascal (1971)
- Ada 83 (1983)
- Ada 95 (1995)

- Bir dilin sözdizimini anlatmak amacıyla kullanılan bir araç vardır. BNF (Backus-Naur Form) **metadili** böyle bir araçtır.
- Anlam tanımlama için böyle bir dil yoktur.



Grammar Nedir?

Example

```
Statement = "if" "(" Condition ")" Statement ["else" Statement].
```

4 bileşenden oluşur

terminal symbols

atomik

"if", ">=", ident, number, ...

nonterminal symbols

Sözdizim değişkenleri

Statement, Expr, Type, ...

productions

Nonterminallerin çözümü

Statement = Designator "=" Expr ";".
Designator = ident ["."] ident].
...

start symbol

Başlangıç nonterminalı

begin

Backus-Naur Form (BNF)-CFG

– CFG

$$\begin{aligned} \textit{expression} &\rightarrow \textit{identifier} \mid \textit{number} \mid - \textit{expression} \\ &\mid (\textit{expression}) \\ &\mid \textit{expression operator expression} \\ \textit{operator} &\rightarrow + \mid - \mid * \mid / \end{aligned}$$

– BNF

$$\begin{aligned} \langle \textit{expression} \rangle &\rightarrow \langle \textit{identifier} \rangle \mid \langle \textit{number} \rangle \mid - \langle \textit{expression} \rangle \\ &\mid (\langle \textit{expression} \rangle) \\ &\mid \langle \textit{expression} \rangle \langle \textit{operator} \rangle \langle \textit{expression} \rangle \\ \langle \textit{operator} \rangle &\rightarrow + \mid - \mid * \mid / \end{aligned}$$

EBNF Notation

Extended Backus-Naur form

John Backus: developed the first Fortran compiler

Peter Naur: edited the Algol60 report

| <i>symbol</i> | <i>meaning</i> | <i>examples</i> |
|---------------|------------------------|------------------------------------|
| string | | "=", "while" |
| name | | ident, Statement |
| = | | A = b c d . |
| . | | |
| | separates alternatives | a b c \odot a or b or c |
| (...) | groups alternatives | a (b c) \odot ab ac |
| [...] | optional part | [a] b \odot ab b |
| {...} | repetitive part | { a } b \odot b ab aab ... |

EBNF

<seçimlik_deyim> -> **If** (**<mantıksal>**) **<deyim>** [**else** **<deyim>**];

BNF

<seçimlik_deyim> -> **If** (**<mantıksal>**) **<deyim>**

<seçimlik_deyim> -> **If** (**<mantıksal>**) **<deyim>** **else** **<deyim>**;

Değiştirme (*alternation*) |

EBNF

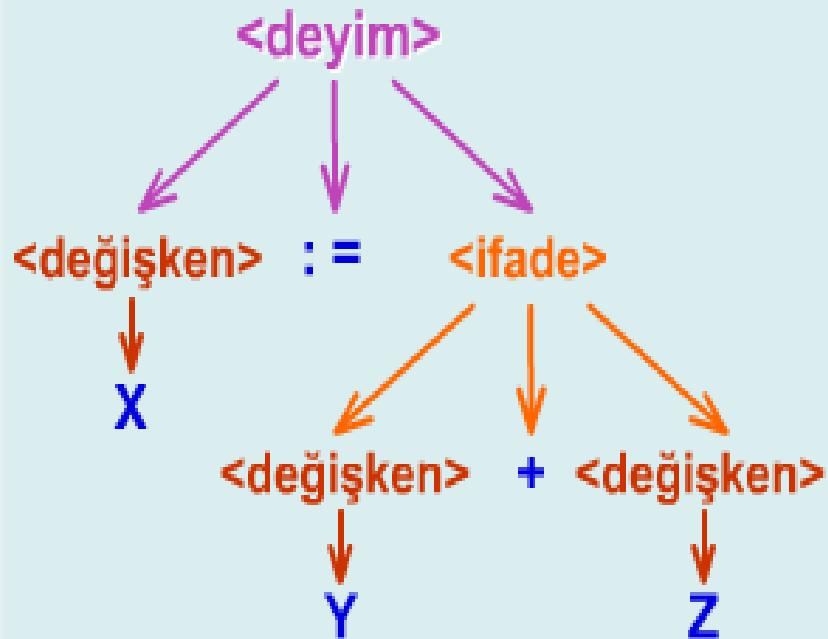
<for_deyimi>->for<değişken> := <ifade> (to | down to) <ifade> do <deyim>

BNF

<for_deyimi>->for<değişken> := <ifade> (to) <ifade> do <deyim>

<for_deyimi>->for<değişken> := <ifade> (down to) <ifade> do <deyim>

Gramerler ve Türetimler



<program> -> begin <deyim_listesi> end

<deyim_listesi> -> <deyim>
| <deyim>; <deyim_listesi>

<deyim> -> <değişken> := <ifade>

<ifade> -> <değişken> + <değişken>
| <değişken>

<değişken> -> X | Y | Z

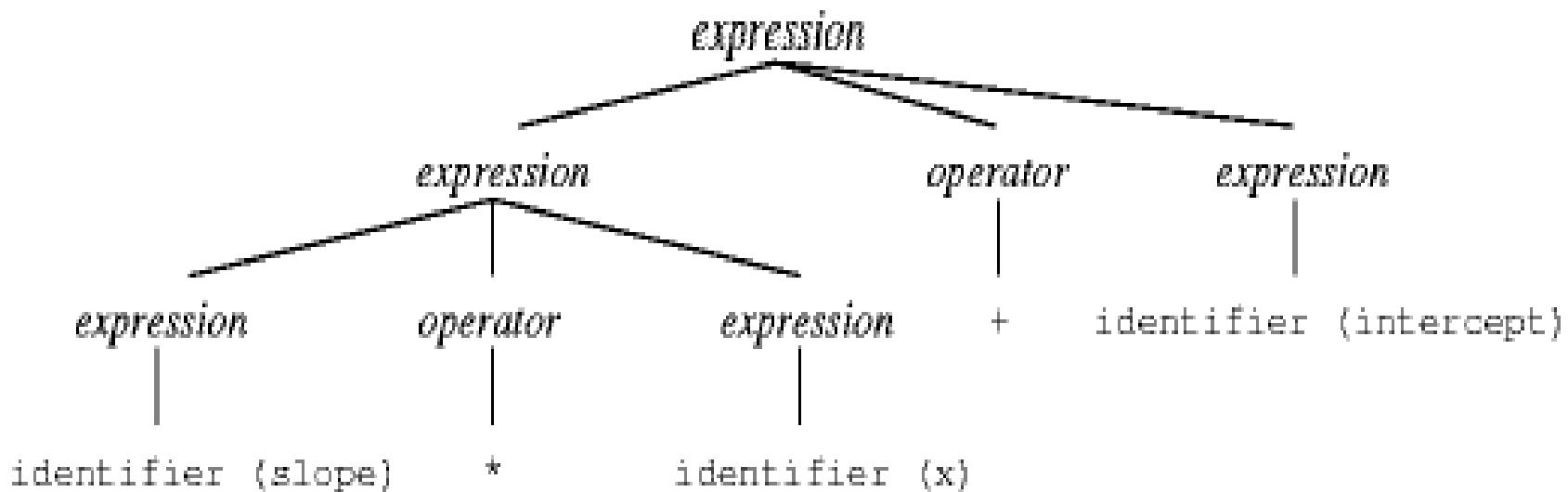
Örnek

- $\text{expression} \rightarrow \text{identifier} \mid \text{number} \mid -\text{expression}$
 $\mid (\text{expression})$
 $\mid \text{expression operator expression}$
 $\text{operator} \rightarrow + \mid - \mid * \mid /$

slope * x + intercept ifadesini
turetelim.

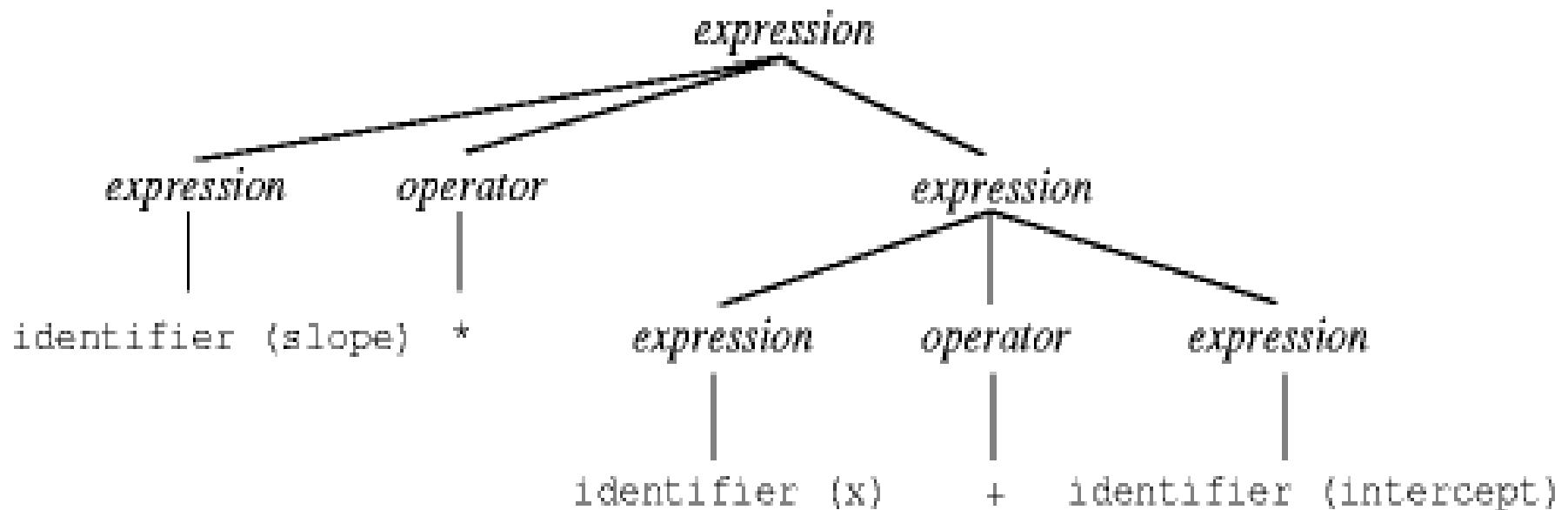
Ayrıştırma Ağacı(Parse Trees)

- Türetimin grafik gösterimi



Belirsiz Gramer

- Bir gramerde aynı ifade için alternatif ayrıştırma ağacı bulunuyorsa bu gramer belirsizdir (ambiguous)



Belirsizlik kaldırılmalıdır.

$$10 - 4 - 3 == (10 - 4) - 3$$

$$3 + 4 * 5 == 3 + (4 * 5)$$

(1) *expression* \rightarrow *term* | *expression add_op term*

(2) *term* \rightarrow *factor* | *term mult_op factor*

(3) *factor* \rightarrow *identifier* | *number* | - *factor* | (*expression*)

(4) *add_op* \rightarrow + | -

(5) *mult_op* \rightarrow * | /

DİL ÇEVİRİMİ

- Yüksek düzeyli bir dilde yazılmış bir program ancak makine diline çevrilerek bir bilgisayarda çalıştırılabilir.

Sözdizim Analizi (Syntax Analysis)

- EBNF

```
<pgm>          -> <statement list> $$$  
<stmt list>   -> <stmt list> <stmt> | E  
<stmt>          -> id := <expr> | read <id> | write <expr>  
<expr>          -> <term> | <expr> <add op> <term>  
<term>          -> <factor> | <term> <mult op> <factor>  
<factor>        -> ( <expr> ) | id | literal  
<add op>        -> + | -  
<mult op>       -> * | /
```

Sözdizim Analizi (Syntax Analysis)

- EBNF

```
<pgm>          -> <statement list> $$$  
<stmt list>   -> <stmt list> <stmt> | E  
<stmt>          -> id := <expr> | read <id> | write <expr>  
<expr>          -> <term> | <expr> <add op> <term>  
<term>          -> <factor> | <term> <mult op> <factor>  
<factor>        -> ( <expr> ) | id | literal  
<add op>        -> + | -  
<mult op>       -> * | /
```

Özel Kelimeler

- Özel kelimeler, bir programlama dilindeki temel yapılar tarafından kullanılan kelimeleri göstermektedir.
- A)Anahtar kelimeler(Keywords)
- B)Ayrılmış kelimeler(Reserved words)

Anahtar Kelime

- Bir anahtar kelime (*keyword*), bir programlama dilinin sadece belirli içeriklerde özel anlam taşıyan kelimelerini göstermektedir.
- Örneğin FORTRAN'da *REAL* kelimesi, bir deyimin başında yer alıp, bir isim tarafından izlenirse, o deyimin tanımlama deyimi olduğunu gösterir. (*REAL apple*)
- Eğer *REAL* kelimesi, atama işlemcisi "=" tarafından izlenirse, bir değişken ismi olarak görülür. *REAL = 10.05* gibi.
- Bu durum dilin okunabilirliğini azaltır.

Ayrılmış Kelime:

- Öte yandan, ayrılmış kelime (*reserved word*), bir programlama dilinde bir isim olarak kullanılamayacak özel kelimeleri göstermektedir.
- C++ dilindeki do, for , while gibi ve
- PASCAL'da procedure, begin, end gibi kelimere ayrılmış kelime denir.

Bağlama ve Kapsam Kavramları

Bağlama(Binding)

- Bir özellikle bir program elemanı arasında ilişki kurulmasına **bağlama (binding)** denir.



Bağlama Zamanı

- Bir programlama dilinde çeşitli bağlamalar farklı zamanlarda gerçekleşebilir.



| | |
|--------------------------------------|----------------------------------|
| int hesap; ... hesap=hesap+10; | |
| Hesap için olası tipler | Dilin tasarım zamanında |
| Hesap değişkeninin tipi | Dilin derlenmesi zamanında |
| Hesap değişkeninin olası değerleri | Derleyici tasarım zamanı |
| Hesabın değeri | Bu deyimin yürütülmesi zamanında |
| + işlemcisinin muhtemel anlamları | Dilin tanımlanması zamanında |
| + işlemcisinin bu deyimdeki anlamı | Derlenme süreci |
| 10 literalinin ara gösterimi | Derleyici tasarımları zamanında |
| Hesap değişkeninin alacağı son değer | Çalışma zamanında |

Bellek Bağlama

- (*allocation*) (*deallocation*) lifetime



etkinlik (activation) kaydı

aynı bellek bölümünün
yeniden kullanılabilmesi

Doğrudan adresleme

Pascal-dispose Java-otomatik
C'deki malloc fonksiyonu
C++ 'daki new işlemcisi

Statik değişkenler, programın yürütülmesi başlamadan bellek hücrelerine bağlanırlar ve bellek hücreleri ile programın çalışması sonlanıncaya kadar bağlı kalırlar. FORTRAN I, II ve FORTRAN IV'de hepsi statik. C, C++ ve Java **static** anahtarını kullanır.

ALGOL 60 ve bu çizgideki diller yiğit dinamik değişkenleri tanımlamaktadır. FORTRAN77 ve FORTRAN90 yerel olarak yiğit dinamik değişkenlere izin vermektedir. Pascal, C ve C+'da, lokal değişkenler, varsayılan olarak yiğit_dinamik değişkenlerdir.

Dışsal yiğin dinamik değişkenlerin bellek yeri bağlaması çalışma zamanında gerçekleşir. Ne kadar bellek gerektiği önceden bilinmez. Çalışma zamanında veriler oldukça belleğe atanır ve bellek yeri yiğin bellekten alınır ve daha sonra yiğin belleğe iade edilir. Bu verilere sadece işaretçi (pointer) değişkenler aracılığıyla ulaşılabilir. Bu değişkenlerin tip bağlaması derleme zamanında, bellek yeri bağlaması ise çalışma zamanında gerçekleşir.

Statik isim kapsam

Değişkenlerin kapsamları, programın metinsel düzeneğine göre, fiziksel yakınlığa göre, belirlenir.

Dinamik Kapsam

Bir ismin kapsamının, altprogramların fiziksel yakınlıklarına göre değil, altprogramların çağrılmama sırasına göre çalışma zamanında belirlenmesi **dinamik kapsam bağlama** olarak adlandırılır.

ALGOL 60'ı izleyen çok sayıda dilde tanımlıdır. Altprogramlar iç içe yuvalanabilir. (C++ ve FORTRAN hariç)

LISP, APL dillerinin ilk sürümleri

1. Altprogramların yuvalanması sonucu gereğinden fazla genel değişken kullanımı olabilir.
2. Bir programda genel olarak tanımlanan değişkenler tüm altprogramlara görünebilir olacakları için güvenilirlik azalmaktadır.

1. Bir altprogramda bir değişkene yapılan başvuru, deyimin her çalışmasında farklı değişkenleri gösterebilir.
2. Programların anlaşılabilirliğini azaltmaktadır

Örnek: Aşağıdaki program parçasının çıkışını

- a) statik kapsam bağlama kurallarına göre
- b) Dinamik kapsam bağlama kurallarına göre bulunuz.

```
int x;

int main() {
    x = 2;
    f();
    g();
}

void f() {
    int x = 3;
    h();
}

void g() {
    int x = 4;
    h();
}

void h() {
    printf("%d\n",x);
}
```

Dinamik kapsam bağlamaya göre çıkış: 3 4

Statik kapsam bağlamaya göre çıkış: 2 2

Ada (exit deyimi)

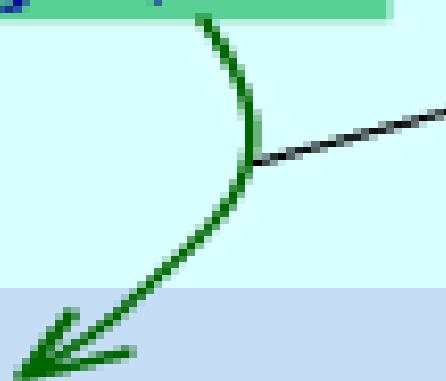
```
loop
```

```
....
```

```
exit when koşul;
```

```
....
```

```
end loop;
```



QuickBASIC'te *exit* deyimi

```
for j=1 to 5
    input miktar
    if miktar < 0 then exit for
    toplam = toplam + miktar
end
```



Neden Altprogram

- Program kodlarının gereksiz yere uzaması önlenir.
- Programın tasarılanmasını kolaylaştırır ve okunabilirliğini artırır.
- Büyük bir programın, daha küçük ve yazılması daha kolay fonksiyonel parçalara bölünür.
- Ekip çalışması
- Altprogramlar birden çok uygulama arasında paylaşılabilir.

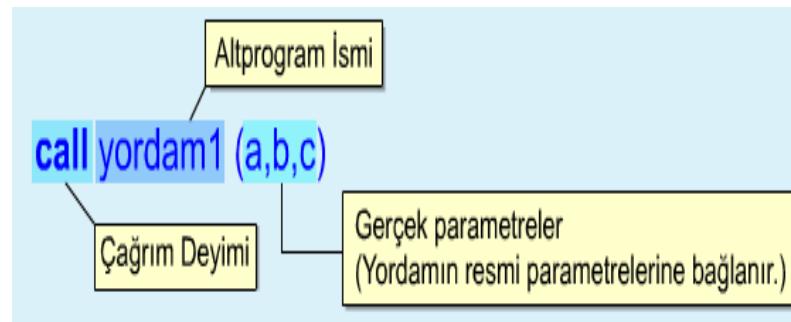
- Bir altprogram bir fonksiyon ise, parametre profiline ek olarak fonksiyonun sonuç değerinin dönüş tipini de içerir ve bu bilgiye **altprogram protokolü** adı verilir.
- C programlama dilinde, altprogramların tanımlandığı deyimler yer alır. Bu deyimlerde altprogramların parametre tipleri belirtilir ve bu deyimlere **prototip** (*prototype*) adı verilir.
- Bir altprogram, bir **yordam** veya bir **fonksiyon** olabilir
- İki altprogram türü arasındaki fark, fonksiyonların çağrıran program birimine bir sonuç değeri döndürmelerinin şart olmasıdır.

procedure ortalama (parametreler);

ortalama (parametreler);

Parametreler

- **gerçek parametrelerin** ve **resmi parametrelerin** ilişkilendirilmesi için kullanılan iki yöntem:
- **konumsal** ve **anahtar kelime parametre** yöntemleridir.
- çağrılm deyiminde, altprogramın ismine ek olarak, yordamın resmi parametreleriyle eşleştirilecek **gerçek (actual) parametreler** de yer alır.
-



Konumsal Parametreler

- Eğer bir yordam çağrılarında gerçek parametreler ile resmi parametreler arasındaki bağlama, parametrelerin çağrım deyimindeki ve yordam başlığındaki konumuna göre yapılıyorsa, bu parametrelere **konumsal (positional) parametre** adı verilir. Birçok programlama dilinde uygulanan bu yöntem, parametre sayısı az olduğu zaman kullanışlıdır.

- Aşağıda görüldüğü gibi bir çağrım deyimi ile etkin duruma geçen *ortalama* yordamında, *c* resmi parametresi, *a* gerçek parametresi ile; *d* resmi parametresi, *b* gerçek parametresi ile bağlılıyorsa, konumsal parametreler yaklaşımı uygulanmıştır.

```
call ortalama(a,b);  
Procedure ortalama(c: integer; d:integer);  
begin  
  ...  
end;
```

| Resmi Parametre | Gerçek Parametre |
|-----------------|------------------------|
| <i>c</i> | \rightarrow <i>a</i> |
| <i>d</i> | \rightarrow <i>b</i> |

Anahtar Kelime Parametre Yöntemi

- Gerçek ve resmi parametreler arasındaki bağlamayı konuma göre belirlemek yerine, her iki parametrenin de ismini belirterek gösterme, **anahtar kelime parametre** yöntemi olarak adlandırılır. Bu durumda çağrımda deyiminde, hem resmi, hem de gerçek parametrelerin isimleri belirtilir.

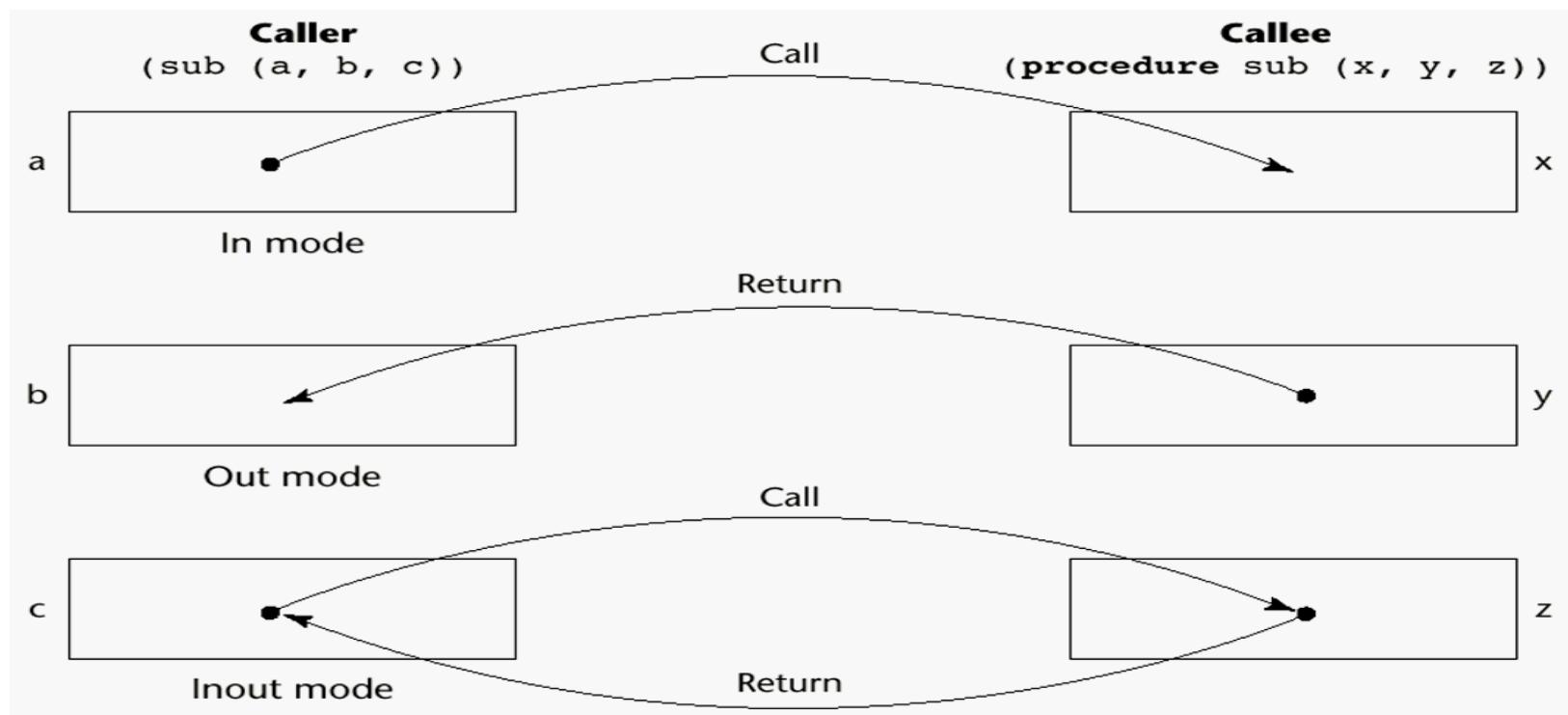
Call (toplam→altntoplasm, liste→dizi, gelir→kazanc);

Anahtar kelime parametre yöntemi, parametre sayısının çok olduğu durumda yararlı bir yöntemdir.

Ada ve FORTRAN 90'da hem anahtar kelime hem de konumsal parametreler yöntemi kullanılabilmektedir.

Parametre Aktarım Yöntemleri

- Bir yordam etkin duruma getirilirken, çağrım deyimindeki gerçek parametreler ile yordamın resmi parametrelerine farklı değerler aktarılabilir.
Gerçek parametreler yordamlara aktarılacak değerleri gösterdikleri için, değişken,sabit veya ifade olabilir. Resmi parametreler ise bu değerleri tutacak bellek yerlerini gösterdikleri için değişken olmak durumundadır.



Parametre Aktarım Yöntemleri (Devam)

- Değer ile çağrıma, sonuç ile çağrıma, değer ve sonuç ile çağrıma yöntemlerinde, gerçek ve formal parametreler arasındaki veri fiziksel olarak kopyalanarak aktarılmakta, başvuru ile çağrıma yöntemiinde ise veri yerine verinin adresi aktarılmaktadır.

Değer ile Çağırma (*Call by Value*)

- Bu yöntemde formal parametre, gerçek parametrenin değeriyle ilklendikten sonra, altprograma yerel bir değişken olarak değerlendirilir.

```
void f(int n)
{   n++;}
int main() {
    int x = 2;
    f(x);
    cout << x; }
```

Sadece gerçek parametreden formal parametreye değer geçisi olduğu için en güvenilir parametre aktarım yöntemidir

Programın çıktısı= 2 olacaktır.

Sonuç ile Çağırma (*Call by Result*)

- Bu yöntemde çağrılmış deyimi ile altprograma bir değer aktarılmazken, gerçek bir parametreye karşı gelen formal parametrenin değeri, altprogram sonunda, denetim yeniden çağrıran programa geçmeden önce, gerçek parametreyi gösteren değişkene aktarılır. (gerçek parametrenin değişken olmalı). Gerçek parametreye karşı gelen resmi parametre, altprogramın çalışması süresince yerel değişkendir.

```
procedure sub1(y: out Integer; z: out Integer) is
```

```
  . . .;  
  sub1(x, x);
```

Değer ve Sonuç ile Çağırma (*Call by Value Result*)

- Değer ile çağrıma ve sonuç ile çağrıma yöntemlerinin birleşimidir.
- Gerçek parametrenin değeri ile karşı gelen formal parametrenin değeri ilklenir ve sonra resmi parametre, altprogramın çalışması süresince yerel değişken gibi davranışır ve altprogram sona erdiğinde formal parametrenin değeri gerçek parametreye aktarılır.

```
int x=0;  
int main()  
{  
    f(x);  
    .....  
}  
void f(int a) {  
    x=3;  
    a++;}
```

x'in son değeri değer-sonuç aktarımına göre 1 olacaktır.

Parametreler için birden çok bellek yeri gereklmesi ve değer kopyalama işlemlerinin zaman almaktadır.

Başvuru ile Çağırma (*Call by Reference*)

- Başvuru ile çağrıma yöntemi de gerçek ve formal parametreler arasında iki yönlü veri aktarımı vardır.
- Altprograma verinin adresi aktarılır. Bu adres aracılığıyla altprogram, çağrıran program ile aynı bellek yerine erişebilir ve gerçek parametre, çağrıran program ve altprogram arasında ortak olarak kullanılır.

```
int x=0;  
int main()  
{  
    f(x);  
    .....  
}  
void f(int a) {  
    x=3;  
    a++;}
```

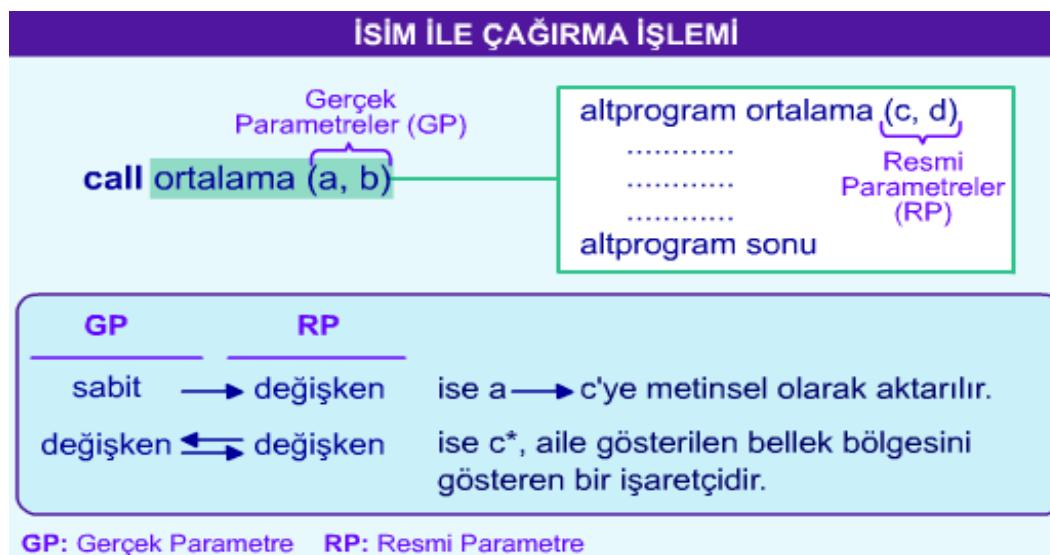
x'in son değeri değer-sonuç aktarımına göre 4 olacaktır.

Hem yer hem de zaman açısından etkindir. Fiziksel kopyalama yoktur.

Formal parametrelere erişim için bir **dolaylı erişim** gerekiyor. Sadece çağrıran programdan altprograma değer aktarılması isteniyorsa, bu yöntemde gerçek parametrenin bellekteki yerine altprogram tarafından ulaşılabilen için, değerinde **istenmeyen değişiklikler** oluşabilir.

İsim ile Çağırma (*Call by Name*)

- Altprogramda gerçek parametreye karşı gelen formal parametrenin bulunduğu her yere metinsel olarak gerçek parametre yerleştirilir.
- Eğer gerçek parametre bir sabit değerse, isim ile çağrıma yöntemi, değer ile çağrıma yöntemi ile aynı şekilde gerçekleşir.
- Eğer gerçek parametre bir değişkense, isim ile çağrıma yöntemi başvuru ile çağrıma yöntemi ile aynı şekilde gerçekleşir.
- İsim ile çağrıma yönteminin gerçekleştirilmesi güçtür ve kullanıldığı programların hem yazılmasını hem de okunmasını karmaşıklaştırabilir. Bu nedenle ALGOL 60 ile tanıtılan isim ile çağrıma yöntemi, günümüzde popüler olan programlama dillerinde uygulanmamaktadır.



| İsim | Referans | Değer-Sonuç | Sonuç | Değer | Programlama Dili |
|------|----------|-------------|-------|----------|--------------------|
| | | | | | FORTTRAN IV |
| | X | | | | Fortran 77 |
| | | X | | | ALGOL 60 |
| X | | | | seçimlik | ALGOL W |
| | | X | | | C++ |
| | X | | | | PASCAL, Modula2 |
| | | X | X | X | ADA |
| | | | | X | Java |

Eşzamanlılık

(Concurrency)

- Programlama dillerindeki eş zamanlılık kavramı ile bilgisayar donanımındaki paralel çalışma birbirinden bağımsız kavamlardır.
- Eğer çalışma zamanında üst üste gelme durumu varsa donanım işlemlerinde paralellik oluşur.
- Bir programdaki işlemler eğer paralel olarak işlenebiliyorsa program eş zamanlıdır denilir.
- Eş zamanlılık kavramının karşıtı ise bilirli bir sıraya göre dizilmiş ardışıl işlemlerdir.

Öncelik grafları:

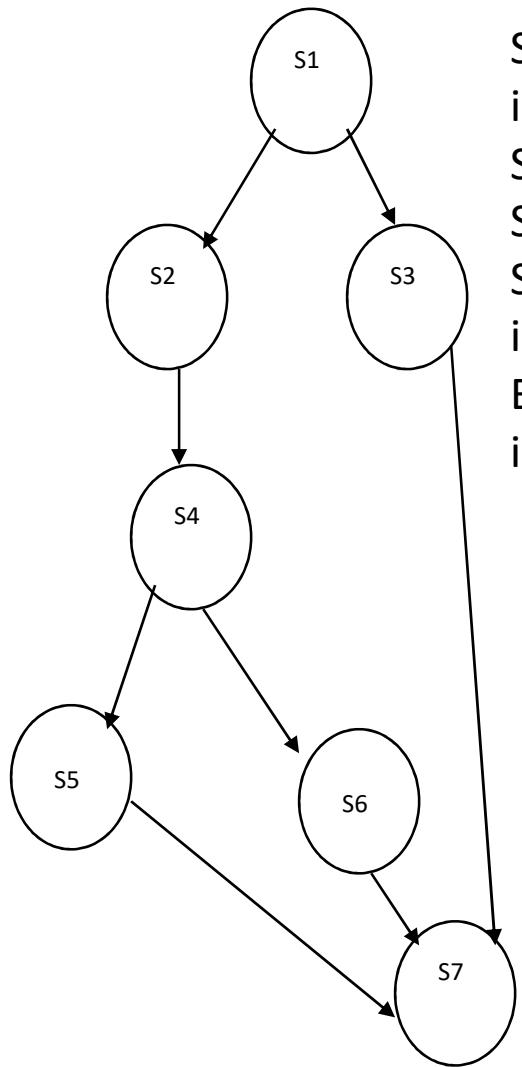
- **a:=x + y;**
- **b:= z + 1;**
- **c:= a - b;**
- **w:= c + 1;**

Burada $c := a - b$ yi hesaplamak için öncelikle a ve b 'ye değer atanması gerekmektedir.

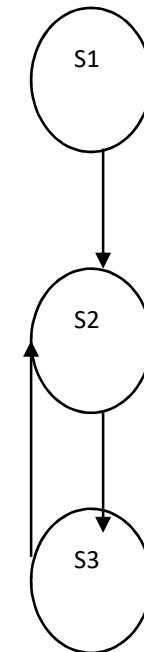
Benzer biçimde $w := c + 1$ ifadesinin sonucu da c 'nin hesaplanmasına bağlıdır.

Diğer taraftan $a := x + y$ ve $b := z + 1$ deyimleri birbirine bağlı değildir. Bu yüzden bu iki deyim birlikte çalıştırılabilir.

Buradan anlaşılıyor ki bir program parçasında değişik deyimler arasında bir öncelik sıralaması yapılabilir. Bu sıralamanın grafik olarak gösterimine öncelik grafi denir. Bir öncelik grafi, her bir düğümü ayrı bir deyimi ifade eden, döngüsel olmayan yönlendirilmiş bir graftır.



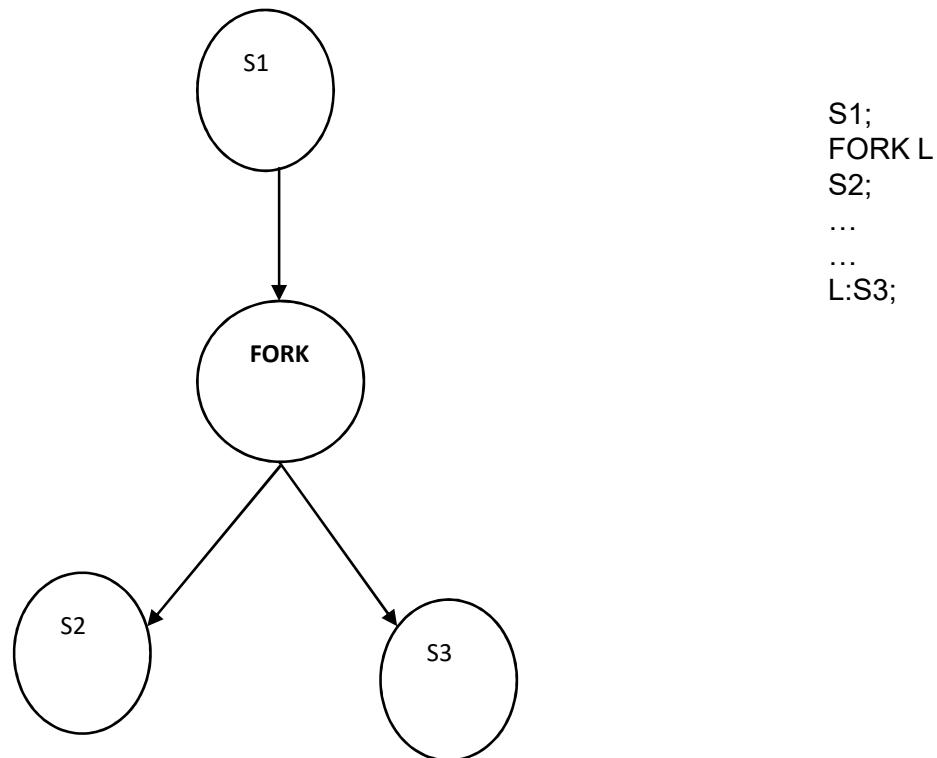
S2 ve S3 deyimleri, S1 tamamlandıktan sonra işletilebilir.
 S4, S2 tamamlandıktan sonra işletilebilir.
 S5 ve S6, S4 tamamlandıktan sonra işletilebilir.
 S7, sadece S5,S6 ve S3 tamamlandıktan sonra işletilebilir.
 Bu örnekte S3 deyimi S2, S4, S5 ve S6 deyimleri ile eş zamanlı olarak çalışabilir.



Bu grafta görüldüğü gibi, S3 sadece S2 tamamlandıktan sonra işletilebilir. S2 deyimi ise sadece S3 tamamlandıktan sonra işletilebilir. Burada açıkça görülmektedir ki bu iki kısıtlamanın her ikisi aynı anda giderilemez. Yani bir programın akışını ifade eden öncelik grafında döngü içermemelidir.

FORK ve JOIN Yapıları:

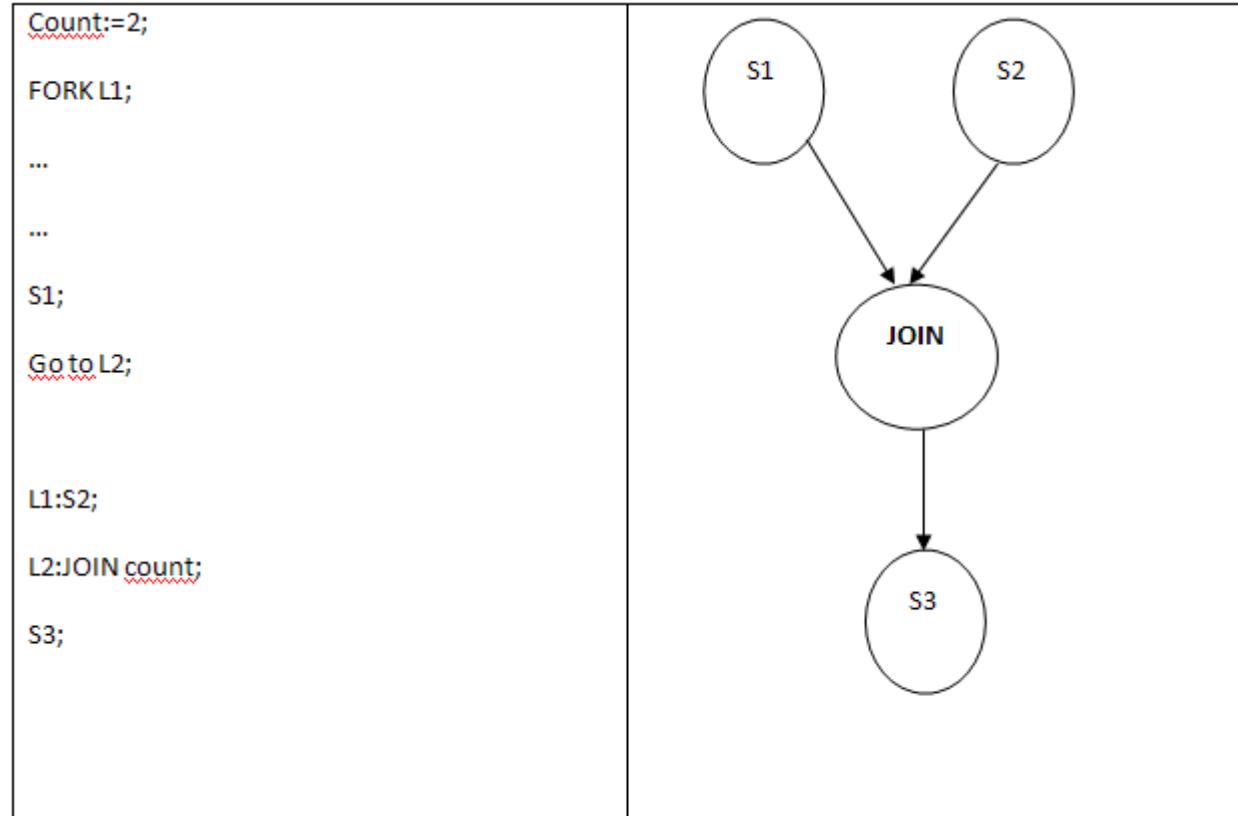
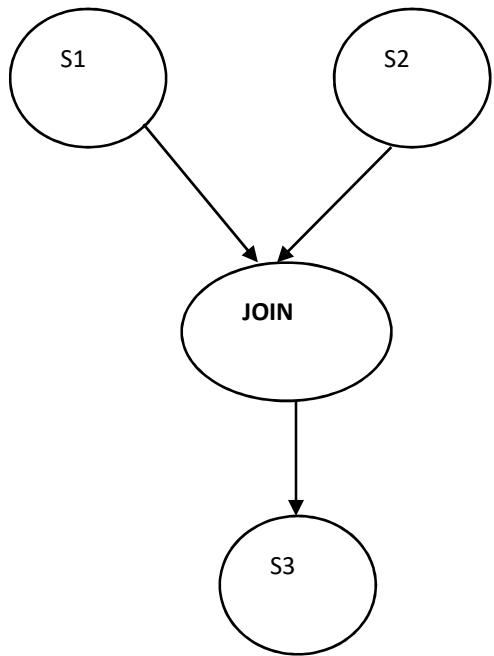
- FORK ve JOIN yapıları eş zamanlılığı tanımlayan ilk programlama dili notasyonlarından biridir. Aşağıdaki öncelik grafi bu komutlardan FORK yapısını ifade etmektedir.



Burada eş zamanlı işlemlerden birisi L etiketi ile gösterilen deyimlerden başlarken diğer FORK komutunu izleyen deyimlerin işlenmesi ile devam eder.

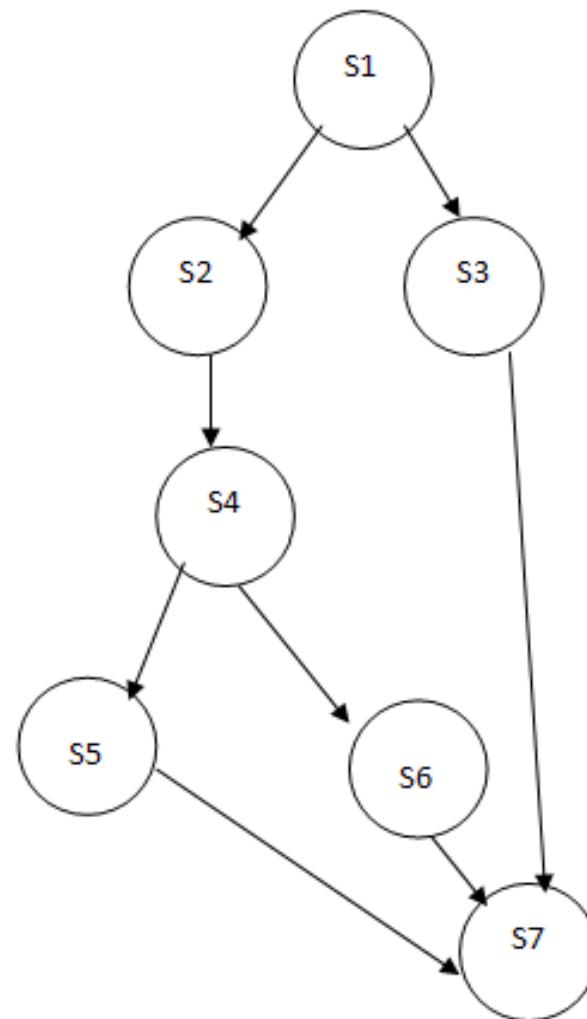
FORK L deyimi işletildiği zaman S3'de yeni bir hesaplama başlar. Bu yeni hesaplama S2'de devam eden eski hesaplama ile eş zamanlı olarak işletilir.

JOIN komutu iki eş zamanlı hesaplamayı tekrar birleştirir. JOIN komutunun öncelik grafi karşılığı aşağıda verilmiştir.



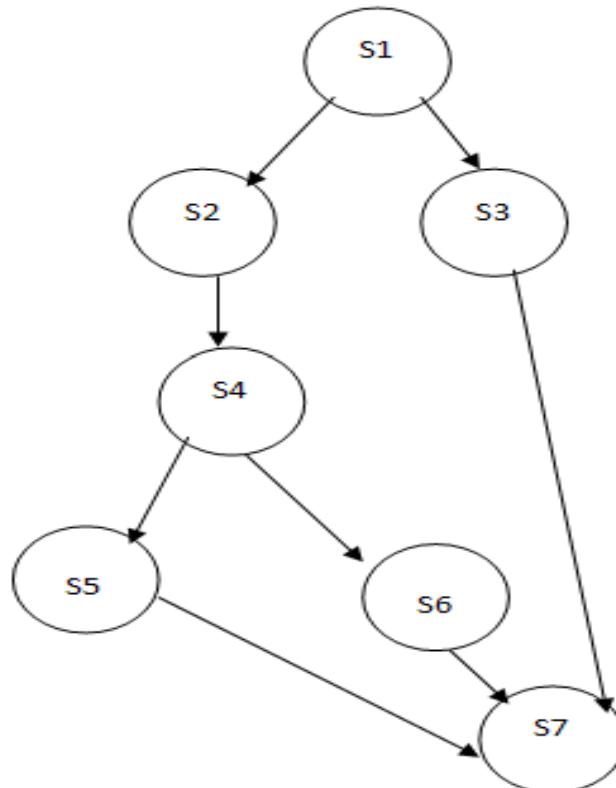
Örnek:

```
S1  
  
Count:=3;  
  
FORK L1;  
  
S2;  
  
S4;  
  
FORK L2;  
  
S5;  
  
Goto L3;  
  
L2:S6;  
  
Goto L3;  
  
L1:S3;  
  
L3: JOIN count;  
  
S7;
```



Örnek

```
S1;  
Parbegin  
S3;  
Begin  
S2;  
S4;  
Parbegin  
S5;  
S6;  
Parend;  
End;  
Parend;  
S7;
```



Emir Esaslı Programlama Yönteminin Değerlendirmesi

- “Böl ve yönet” prensibine dayanır. Amaç büyük programları küçük parçalara bölgerek yazılım geliştirme işini kolaylaştırmaktır.
- Ancak yazılımların karmaşıklıkları sadece boyutlarından kaynaklanmaz. Küçük problemler de karmaşık olabilir.
- Gerçek dünyadaki sistemler sadece fonksiyonlardan oluşmaz. Dolayısıyla emir esaslı yaklaşımda karmaşık bir problemin gerçeğe yakın bir modelini bilgisayarda oluşturmak zordur.
- Tasarım aşamasında verilerden çok fonksiyonlara odaklanıldığından hatalar nedeniyle veri güvenliği tehlikeye girebilmektedir.
- Kullanıcılar kendi veri tiplerini çok güçlü biçimde tanımlayamazlar.
- Programın güncellenmesi zordur.
- İşleve dayalı yöntemi de kullanarak kaliteli programlar yazmak mümkün değildir.

Ancak nesneye dayalı yöntem kaliteli programların oluşturulması için programcılara daha çok olanak sağlamaktadır ve yukarıda açıklanan sakıncaları önleyecek yapılara sahiptir.

Yöntemin Değerlendirmesi:

- Gerçek dünya nesnelerden oluştugundan bu yöntem ile sistemin daha gerçekçi bir modeli oluşturulabilir. Programın okunabilirliği güçlenir.
- Nesne modellerinin içindeki veriler sadece üye fonksiyonların erişebileceği şekilde düzenlenebilirler. Veri saklama (data hiding) adı verilen bu özellik sayesinde verilerin herhangi bir fonksiyon tarafından değiştirilmesi önlenir.
- Programcılar kendi veri tiplerini yaratabilirler.
- Bir nesne modeli oluşturuktan sonra bu modeli çeşitli şekillerde defalarca kullanmak mümkündür (reusability).
- Programları güncellemek daha kolaydır.
- Nesneye dayalı yöntem takım çalışmaları için uygundur.

NESNEYE YÖNELİK PROGRAMLAMA DİLLERİ

- Programlama dilleri literatüründe sınıf ve alt sınıf kavramını ilk olarak SIMULA67dili tanıtmıştır.
- Smalltalk, Eiffel, ADA95 ve Java dillerinde tüm veriler nesne şeklindedir. Yani bu diller tam nesne yönelimlidir.
- C++ ise hem emir esaslı paradigmayı hem de nesneye yönelik paradigmayı destekler.

Smalltalk

- SIMULA67'de tanıtılan fikirler, Smalltalk ile güçlenmiş ve Smalltalk ile nesne yönelimli dil popüler hale gelmiştir
- Smalltalk'ta bir program sadece kalıtım hiyerarşisi içinde düzenlenmiş birbirleriyle mesajlar ile etkileşen nesne sınıflarından oluşabilir. Smalltalk'ta tüm veriler nesnelerle gösterilmek zorunda olduğu için tam nesneye yönelik bir programlama dili olarak nitelendirilir.
- Smalltalk dilinde bütün bağlamlar dinamik olarak gerçekleşir.
- Smalltalk'ta sınıfların sadece tek üst sınıfı bulunabilir (tekli kalıtım modeli).

static, abstract ve *final* tanımlayıcıları

- *static* özelliğindeki bir veri sahası, bir sınıfın tüm örneklerinde paylaşılır. *static* bir metod, sınıfın bir örneği yaratılmadan da çağrılabılır ve *static* bir metod, bir alt sınıfta yeniden tanımlanamaz.
- *abstract* olarak tanımlanmış bir sınıf örneklenemez ve sadece üst sınıf olabilir. Bir *abstract* metod ise bir alt sınıf tarafından gerçekleştirilmek zorundadır.
- *final* olarak tanımlanmış bir sınıfın alt sınıfları tanımlanamaz ve *final* olarak tanımlanmış bir metod, herhangi bir alt sınıfta yeniden tanımlanamaz.

Fonksiyonel Programlama

LISP, ML, Haskel, Scheme

- Emir esaslı dillerin tasarıımı doğrudan doğruya von Neumann mimarisine dayanır.
- Bir imperative dilde, işlemler yapılır ve sonuçlar daha sonra kullanım için değişkenlerde(variables) tutulur. Emir esaslı dillerde değişkenlerin yönetimi karmaşıklığa yol açar.
- Fonksiyonel dillerin tasarıımı Matematiksel Fonksiyonlara dayalıdır ve değişkenler(variables), matematikte olduğu gibi gerekli değildir. Kullanıcıya da yakın olan sağlam bir teorik temele sahiptir.
- Fonksiyonel programlamada , bir fonksiyon aynı parametreler verildiğinde daima aynı sonucu üretir (referential transparency).

Haskell

- Kuvvetli tipli, statik kapsam bağlamalı ve tip yorumlamalı bir dildir.
- Tam olarak fonksiyonel bir dildir. (değişkenler yoktur, atama ifadeleri yoktur, hiçbir çeşit yan etki yoktur).
- Tembel değerlendirme(**lazy evaluation**) kullanır (değer gerekmemiş olduğu sürece hiçbir alt-ifadeyi değerlendirmeme)
- Liste kapsamları(**list comprehensions**), sonsuz listelerle çalışmaya izin verir

Örnekler

1. **fib 0 = 1**

fib 1 = 1

fib (n + 2) = fib (n + 1)
+ fib n

2.

fact n

| **n == 0 = 1**

| **n > 0 = n * fact (n - 1)**

3. Liste işlemleri

– Liste gösterimi:

– `directions = ["north", "south", "east", "west"]`

– Uzunluk(Length): #

#directions = 4

– .. operatorü ile aritmetik seriler

`[2, 4..10]` gösterimi `2, 4, 6, 8, 10` olarak değerlendirilir.

Örnekler devam

3. Liste işlemleri(devam)

- ++ ile zincirleme(Catenation)

[10, 30] ++ [50, 70] → [10, 30, 50, 70]

- :operatörü yoluyla

1: [3, 5, 7] → [1, 3, 5, 7]

```
product [] = 1
product (a:x) = a * product x

fact n = product [1..n]
```

Haskell örnekler (devam)

4. Liste kapsamları:küme gösterimi

```
[n * n | n ← [1..10]]
```

ilk 10 pozitif tamsayının karelerinden oluşan
bir liste tanımlar

```
factors n = [i | i ← [1..n `div`  
2],  
              n `mod` i == 0]
```

Bu fonksiyon verilen parametrenin bütün
çarpanlarını hesaplar

Haskell örnekleri

- Quicksort(Hızlı Sıralama):

```
sort [] = []
```

```
sort (a:x) = sort [b | b <- x; b
<= a]
```

```
++ [a] ++
```

```
sort [b | b <- x; b > a]
```

Haskell örnekleri (devam)

5. Tembel değerlendirme(Lazy evaluation)

```
positives = [0..]
```

```
squares = [n * n | n ← [0..]]
```

(sadece gerekli olanları hesapla)

```
member squares 16
```

True döndürür

1. Aşağıdakilerden hangisi Mantıksal Programlama ile ilişkilidir?
 a) Lambda Kalkülüs b) Kalitim c) BNF
 d) Predicate Kalkülüs e) Fortran

2. Aşağıdakilerden hangisi Mantıksal Paradigma içerisinde örnek gösterilebilecek bir programlama dilidir?
 a) Prolog b) Common Lisp c) Python
 d) ProLogic e) Basic

3. Aşağıdakilerden hangisi Fonksiyonel paradigmının avantaj veya avantajlarından?
 I. Basit Semantik a) Yalnız I b) I-II-III
 II. Basit Sözdizim c) I-II-IV d) Yalnız IV
 III. Yüksek Verim e) III-IV

IV. Otomatik Eşzamanlılık

4. Aşağıdakilerden hangisi nesneye yönelik paradigm ile alakalı **değildir**?
 a) Kalitim b) Geç Bağlama (Late Binding)
 c) Çok Biçimlilik d) Kapsüleme

5. Aşağıdakilerden hangisi Nesneye Yönelik Programlamanın özelliklerinden **değildir**?
 a) Örnekler sınıflardan kurulum yoluyla oluşturulur.
 b) Kapsül haline getirilmiş bir nesnenin alanları dışa kapalıdır.
 c) Kalitim alınan üst sınıftır.
 d) public olarak tanımlanmamış bir sınıf türetilmez.

6. istisnai durumların önlenmesi bakımından C++ dili aşağıdaki terimlerden hangisi veya hangilerini destekler?
 I.try
 II.catch
 III.finally
 IV.throw
 V.throws

- a) I ve II b) I, II ve III c) I, II, IV ve V
 d) I, II ve IV e) I, II, III ve IV

7. Aşağıdakilerden hangisi Yapısal Programlamanın ilgilendiği başlıklardır?
 I. Alt Programlama a) Yalnız I b) I-II-III
 II. Kontrol Yapıları c) II-III d) III-IV
 III. Döngü Yapıları e) III-IV
 IV. Kalitim

8. Aşağıdaki Programlama dillerinin değerlendirme kriterlerinden hangisinin desteği arttığında güvenliğin azalabileceği söyleylenebilir?
 a) Esneklik b) İfade Gücü c) Taşınabilirlik
 d) I/O Kolaylığı e) Yapısallık

9. Alt indis sıfır olan ve her bir elemanın sözcük uzunluğu c olan bir A dizisinin k. indisinin adresi hangi şıkta doğru olarak verilmiştir?
 a) Adres(A[k-1])=Adres(A[0])+(k-1)*c
 b) Adres(A[k])=Adres(A[0])+(k-1)*c
 c) Adres(A[k])=Adres(A[0])+(k)*c
 d) Adres(A[k-1])=Adres(A[0])+(k)*c
 e) Adres(A[k+1])=Adres(A[0])+(k)*c

10. Derlenme sürecinde aşağıdaki aşamalardan hangisinde regüler ifade, DFA ve gramer gibi araçlarla tanimlanamayan bilgiler içerir?

- a) Optimizasyon b) Ara Kod Dönüşürücü
 c) Lexical Analiz d) Syntax Analiz
 e) Semantic Analiz

11. Hata yakalamada Java dilindeki throw ifadesi C dilindeki aşağıdaki hangi fonksiyon ile ilişkilendirilebilir?
 a) longjmp b) setjmp c) jmp_buf d) ifjmp e) thjmp

```
12. Yandaki kod int v;      void T1(){  

    sırasıyla statik ve int main(){      int v=8;  

    dinamik kapsamları      T2();  

    bağlamaya göre      T1();  

    değerlendirdiğinde      T2();  

    ekran çıktısı ne olur?      void T3(){  

    return 0;      T3(v);  

    a) 77 88      }  

    b) 88 77      }  

    c) 77 77      void T3(int v){  

    d) 88 88      printf("%d",v);  

    e) 77 87      }
```

13. Aşağıdaki dillerin hangisinde Heap bellek bölgesinde oluşan çöpleri otomatik toplayacak bir mekanizma **yoktur**?
 a) Pascal b) Python c) Java d) C# e) Ada

14. Aşağıdaki BNF ifadesi EBNF ile nasıl ifade edilebilir?
 <degisen> ::= <harf> | <degisen><harf> | <degisen><rakam>
 a) <degisen> ::= <harf> <degisen><harf><rakam>
 b) <degisen> ::= <harf> <char> | <rakam>
 c) <degisen> ::= <harf> <harf><rakam>
 d) <degisen> ::= <harf> <char> | <rakam>
 e) <degisen> ::= <harf> <harf><rakam>

15. C++ dilinde void fonk(int x,out int y) { ... } şeklinde tanımlanmış bir fonksiyon ile aynı etkiye yapmak için C++ dilinde nasıl tanımlanmalıdır?

- a) void fonk(int x, int **y){ ... } b) void fonk(int x, int *y){ ... }
 c) void fonk(int x, int &y){ ... } d) void fonk(int x, int &&y){ ... }
 e) void fonk(int x, int y){ ... }

16. Java'nın yandaki parametre I. Değer ile çağrıma çağrıma yöntemlerinden II. Adres ile çağrıma hangisini veya hangilerini III. Referans ile çağrıma desteklediği söyleylenebilir?
 a) Yalnız I b) II-III c) I-IV d) I-III e) I-II

17. Diamond problemi neden kaynaklanabilir?
 a) Kapsüleme b) Basit Sözdizim c) Çoklu Kalitim
 d) Soyut Sınıf e) Geç Bağlama

```
S1:  

FORK L1;  

S2;  

goto L2;  

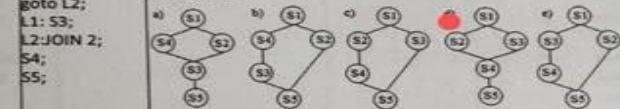
L1: S3;  

L2: JOIN 2;  

S4;  

S5;
```

18. Fork ve Join kullanılarak gerçekleştirilen eş zamanlılık yapısına karşılık gelen öncelik grafi hangisidir?



19. Aşağıdaki c++ koduna karşılık gelen, Lisp kodu aşağıdakilerden hangisidir?

```
int Hadi(int a,int b=10){ return a+b; }  

a)(defun Hadi(a b)  

   (setq b 10)  

   (+ a b))  

b)(defun Hadi(a &optional b)  

   (if (eq b nil) (setq b 10)  

       (+ a b))  

c)(defun Hadi(a b)  

   (if (eq b nil) (setq b 10)  

       (+ a b))  

d)(defun Hadi(&optional a b)  

   (if (eq b nil) (setq b 10)  

       (+ a b))  

e) Tanımlanamaz
```

20. **typedef enum FF{false, true}class;** ifadesi ile C dilinde Java'daki hangi türün benzetimi yapılmış olabilir?
 a) class b) int c) boolean d) char e) byte

| | |
|---|---|
| 8 | 9 |
| 0 | 0 |
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |
| 4 | 4 |
| 5 | 5 |
| 6 | 6 |
| 7 | 7 |
| 8 | 8 |
| 9 | 9 |

AAAAAA

Aşağıdaki ifadelerde boş bırakılan kısımları uygun kelime/kelimelerle dolduruz.

1. Fortran dilinin bilgisayar tarafından anlaşılması için **DERLEYİCİ**'ye ihtiyacı vardı. **(5 puan)**
2. Yazılan bir programın o dile ait olup olmadığını belirleyen kurallar, **SÖZDİZİM (SYNTAX) VE ANLAM (SEMANTICS)** olarak ikiye ayrılabilir. **(5 puan)**
3. Derleme sürecinin başlangıcında derleyiciye verilen yüksek düzeyli bir programlama dili deyimlerini içeren programa, **KAYNAK (SOURCE) PROGRAM** derleme sürecinin sonucunda oluşan makine dilindeki programa ise **AMAÇ (OBJECT) PROGRAM** adı verilir. **(5 puan)**
4. C++ programlama dilinde sınıflardan çoklu miras alma işlemi gerçekleştirilebilir. C#'da sınıflar **ARAYÜZ (INTERFACE)** çoklu kalıtım alabilir. **(5 puan)**
5. Bir ismin kapsamının, altprogramların fiziksel yakınlıklarına göre değil, altprogramların çağrılmaya sırasında çalışma zamanında belirlenmesi **DİNAMİK KAPSAM BAĞLAMA** olarak adlandırılır. **(5 puan)**
6. C dilindeki ondalık sayıları tanımlama için bir BNF yazınız. (123.1112 ve/veya 123E-2 gibi) **(15 puan)**

sayılar → (+|-|ε) rakamlar(noktalı|Eli)

Eli → E (+|-|ε) rakamlar

noktalı→(.)rakamlar

rakamlar → (rakam){rakam}

rakam → 0 | 1 | ... | 9

7) Aşağıda C#'ta yazılmış olan ara yüzü (interface), C++'ta tekrar yazınız. **(10 puan)**

| | |
|---|---|
| public interface IDikdortgen { void DegerleriAta(int x,int y); int Alan(); } | class IDikdortgen { public: virtual void DegerleriAta(int x,int y) = 0; virtual int Alan() = 0; }; |
|---|---|

8) HavaAraci sınıfının başlık dosyasını yazınız. **(10 puan)**

Not: Başlık dosyası ders notlarında belirtilen şekilde olmalıdır.

Araç Sınıfının Yapıcı Metodu: Arac(float hiz, string model, string motor) { ... }

```
#ifndef HAVAARACI
#define HAVAARACI
class HavaAraci : public Arac
{
    private:
        string kaptan_adi;
    protected:
        string KaptanKim();
    public:
        void Alcal(float);
        void Yuksel(float);
        bool TekerlerKapali();
        HavaAraci(float hz, string md, string mt, string
isim):Arac(hz,md,mt) { kaptan_adi=isim; }
};

#endif
```

Ad/Soyad:
Numara:
Dersi Aldığı Akademisyen:

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ
PROGRAMLAMA DİLLERİNİN PRENSİPLERİ
2012 – 2013 BAHAR FİNAL

21.05.2013
Süre: 65 dakika

9) Aşağıdaki goto içeren C++ kodunu goto kullanmadan yazınız. Yazdığınız kod aynı işlemi yapmalıdır. (12 puan)

```
int main()
{
    int x=0,i;
    string isim="ahmet";
    mm:
    i=0;
    x++;
    if(x == 10) goto cc;
    nn:
    cout<<isim<<" "<<x<<endl;
    i++;
    if(i == 10) goto mm;
    else goto nn;
    cc:
    return 0;
}
```

```
int main()
{
    string isim="ahmet";
    for(int x=1;x<10;x++)
    {
        for(int i=0;i<10;i++)
        {
            cout<<isim<<" "<<x<<endl;
        }
    }
    return 0;
}
```

10) Lisp programlama dilinde parametre olarak verilen 3 sayının en küçüğünü döndüren fonksiyonu yazınız. Eşitlik durumu da göz önünde bulundurulmalıdır. (10 puan)

```
(Defun EnKucuk(x y z)
  (if (and (<= x y) (<= x z)) x
      (if (and (<= y x) (<= y z)) y
          z
      )
  )
)
```

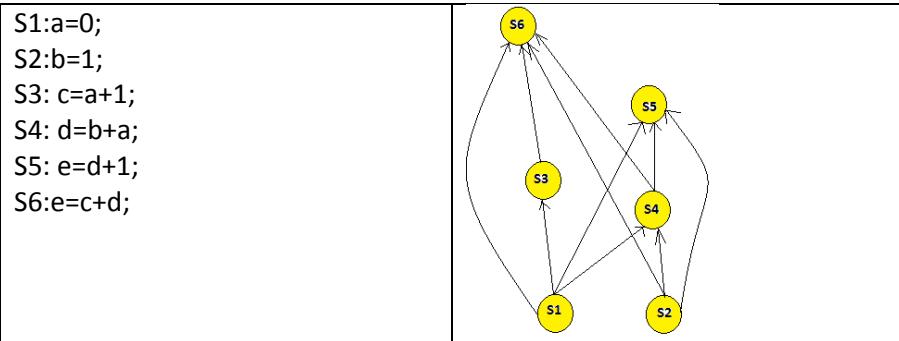
11) Lisp programlama dilinde yazılmış olan aşağıdaki FF fonksiyonu çağrıldığında ekrana ne yazacağını kalın ile yazılmış print komutlarının karşısına yazınız. (6 puan)

```
(Defun Topla(x y)
  (+ x y)
  (incf x)
)
(Defun Cikar(x y)
  (- x y)
  (decf x)
)
(Defun FF()
  (setq sonuc (+ #C(2 3) #C(1 5)))
  (print sonuc)
  (print (Topla 4 3))
  (print (+ (Topla 2 2) (Cikar 5 1)))
  "___"
)
```

| |
|---------|
| #C(3 8) |
| 5 |
| 7 |

12) Aşağıda verilen deyimlerin eş zamanlı çalışmasını kontrol etmek üzere tasarlanacak öncelik grafini çiziniz. (12 puan)

S1:a=0;
S2:b=1;
S3: c=a+1;
S4: d=b+a;
S5: e=d+1;
S6:e=c+d;



1- Aşağıdaki öncelik ve birleşme özelliği tablosuna göre verilen işlemlerin değerlendirme sırasını yazınız(15 p). (Ö.C. 2,3)

| | | |
|----------|-------------|---------------------|
| Öncelik | En Yüksek | * , /, not |
| | | + , - , &, mod |
| | | - (Tekli) |
| | | =, /=, <, <=, >, >= |
| | En Düşük | and |
| Birleşme | Soldan Sağa | OR, xor |

- $a^*(b-1)/c \bmod d$
 1. **c mod d**
 2. **(b - 1)**
 3. **1./2.**
 4. **a*3.**
- $(a-b)/c \& (d^*e/a-3)$
 1. **a-3**
 2. **e/1.**
 3. **d*2.**
 4. **c&3.**
 5. **(a-b)**
 6. **5./4.**
- $-a$ OR C = and e

Bu sık hatalı verildiği için herkesin doğru kabul edilmiştir.

2- İç içe altprogramlara izin veren bir programlama dilinin olduğu varsayılar ise, Bir altprogram bildirimini **subprogram** sözcüğüyle başlar ve **end** sözcüğü ile biter. **decl** tamsayı değişken tanımlar ve **call** bir altprogramı çağırır. **output** ise veriyi ekranaya yazar (15 p). (Ö.C. 3, 4)

Subprogram main

```
decl i
subprogram sub1
    output(i)
    i=4
end
subprogram sub2
    dec i
    subprogram sub3
        call sub1
        output(i)
    end
    i=3
    call sub3
    output(i)
end
i=2
call sub2
output(i)
end
```

main işletiminden başlayarak programı tarayınız. Değişkenlerin değerlerini ve ekran çıktılarını

- a) Durağan kapsam bağlama (static scoping) **2, 3, 3, 4**
- b) Dinamik Kapsam Bağlama (Dynamic Scoping) **3, 4, 4, 2**

İçin ayrı ayrı inceleyiniz.

3- Aşağıda C++'ta yazılmış olan kodu **global değişken kullanmadan** yeniden yazınız. Kullanılan fonksiyon sayısı **değişmemelidir**.

Program, değiştirilen kodda da **aynı sonucu** üretmelidir. (15 p) (Ö.C. 3)

| | |
|--|---|
| <pre>int uzunluk=7; int dizi[] = {15, 82, 16, 90, 2, 12, 100}; int eleman; void TersCevir() { int tmp; for (int i = 0; i < uzunluk/2; i++) { tmp = dizi[uzunluk-i-1]; dizi[uzunluk-i-1] = dizi[i]; dizi[i] = tmp; } } void Eleman(int indeks){ eleman = dizi[indeks]; } int main(){ TersCevir(); for(int i=0;i<uzunluk;i++){ Eleman(i); cout<<eleman<<" "; } return 0; }</pre> | <pre>void TersCevir(int *dizi,int uzunluk) { int tmp; for (int i = 0; i < uzunluk/2; i++) { tmp = dizi[uzunluk-i-1]; dizi[uzunluk-i-1] = dizi[i]; dizi[i] = tmp; } } int Eleman(int *dizi,int indeks){ return dizi[indeks]; } int main(){ int dizi[] = {15, 82, 16, 90, 2, 12, 100}; TersCevir(dizi,7); for(int i=0;i<7;i++){ cout<<Eleman(dizi,i)<<" "; } return 0; }</pre> |
|--|---|

4- Aritmetik işlemler (+, -, /, *) için bir BNF yazınız (10 p). (Ö.C. 3,4)

```
<expr> -> <expr> + <expr>
<expr> -> <expr> - <expr>
<expr> -> <expr> * <expr>
<expr> -> <expr> / <expr>
<expr> -> (<expr>)
<expr> -> <id>
```

5- Aşağıdaki C++ kodu çalıştırıldığında ekrana ne yazacağını yanındaki boşluğa yazınız (10 p). (Ö.C. 3)

| | |
|--|---------------|
| <pre>int main(){ int x = 100; int y = 1000000; int a = sizeof x; int b = sizeof y; if(a > b) cout<<"Sakarya"; if(b > a) cout<<"İstanbul"; else cout<<"Ankara"; return 0; }</pre> | <p>Ankara</p> |
|--|---------------|

Ad/Soyad:
Numara:

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ
PROGRAMLAMA DİLLERİNİN PRENSİPLERİ II. ÖĞRETİM A ve Karma
2014 – 2015 Bahar Dönemi Vize Sınavı Cevapları

07.04.2015
Süre: 70 dakika

6- Aşağıda Java'da yazılmış for döngüsünü yine Java'daki foreach yapısına dönüştürünüz. (15 p) (Ö.C. 3,4)

```
String cumle = "Bu dersten herhalde kaldım.";  
for(int i=0;i<cumle.length();i++)  
    System.out.print(cumle.charAt(i)+" ");  
  
System.out.println();
```

```
String cumle = "Bu dersten herhalde kaldım.";  
for(char c : cumle.toCharArray())  
    System.out.print(c+" ");  
System.out.println();
```

7. Java gibi yüksek seviyeli bir dil erişilemeyen satırı izin vermez. Erişilemeyen satırı birkaç satır kod yazarak gösteriniz. (10 p) (Ö.C. 2)

```
public int Topla(int x,int y){  
    return x+y;  
    int sonuc = x+y; // Erişilemeyen satır hatası  
}
```

// Gibi birçok örnek verilebilir.

8. Aşağıda C++'ta yazılmış kodu, Java'da yazınız. Vereceği çıktı aynı olmalıdır (10 p). (Ö.C. 3,4)

```
#include <iostream>  
using namespace std;  
int main(){  
    for(int i=1;i<=9;i++){  
        inner:  
        for(int j=1;j<=9;j++){  
            if(i*j >= 10) goto outer;  
            cout<<" "<<i*j;  
        }  
    }  
    outer:  
    cout<<endl;  
    return 0;  
}
```

```
outer:  
for(int i=1;i<=9;i++){  
    inner:  
    for(int j=1;j<=9;j++){  
        if(i*j >= 10) break outer;  
        System.out.print(" "+i*j);  
    }  
    System.out.println();
```

Önemli: Her sayfada mutlaka adınız yazmalıdır. Her soruyu altında veya yanında bırakılan boş yerlere cevaplayınız. Ek kâğıt verilmeyecektir.

Ad/Soyad:
Numara:

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ
PROGRAMLAMA DİLLERİNİN PRENSİPLERİ II. ÖĞRETİM A ve Karma
2015 – 2016 Bahar Dönemi Vize Sınavı

18.04.2016
Süre: 60 dakika

1- Aşağıdaki kod düşünüldüğünde main işletiminden başlayarak Durağan ve dinamik kapsam bağlama açısından ekran çıktısını main(3) ve main(0) çağrımları için ayrı ayrı yazınız? (14 p) (Ö.C. 3, 4)

```
int i=0;  
void A(){  
    print i;  
}  
void B(){  
    int i;  
    i=2;  
    A();  
}  
int main(int a){  
    if(a>0) B();  
    else A();  
}
```

2- Aşağıdaki tanımlar için EBNF'leri yazınız. (16 p) (Ö.C. 3,4)

- a.) Java programlama dilinde sınıfın sadece başlık tanımını yapan EBNF'yi yazınız.
Başlık Tanımı demek sınıf tanımında { işaretine kadar olan kısımdır.
- b.) C++ dilinde switch deyimi için EBNF yazınız.

3- Öncelik ve birleşme kurallarının aşağıdaki gibi olduğunu kabul ederek verilen işlemlerin hangi sırada yapılacağını yazınız. (20 p) (Ö.C. 3,4)

| | | | |
|---------|-----------|----------------------|----------|
| Öncelik | En Yüksek | * | , /, not |
| | | +, -, &, mod | |
| | | - (unary) | |
| | | =, /=, <, <=, >, >=, | > |
| | | and | |
| | En Düşük | or, xor | |

Birleşme: Soldan sağa

- a) $a * b - 1 + c$
- b) $a * (b - 1) / c \text{ mod } d$
- c) $(a - b) / c \& (d * e / a - 3)$
- d) $\neg a \text{ or } c = d \text{ and } e$

4- Aşağıdaki kodlar çalıştırıldığında ekrana ne yazar (20p)?

| | |
|---|--|
| <pre>public static void main(String[] args) { int x=15; int y=2; double sonuc = (double)(x/y); System.out.println(sonuc); }</pre> | |
| <pre>void FF(int y){ y++; } void fonk(int& x){ x++; FF(x); } int main(){ int sayı=5;</pre> | |

Ad/Soyad:
Numara:

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ
PROGRAMLAMA DİLLERİNİN PRENSİPLERİ II. ÖĞRETİM A ve Karma
2015 – 2016 Bahar Dönemi Vize Sınavı

18.04.2016
Süre: 60 dakika

```
fonk(sayı);
cout<<sayı;
return 0;
}
```

5- Aşağıdaki Java kodunu C++'ta derlenecek şekilde tekrar yazınız (20p).

```
public abstract class GeometrikSekil {
    private String renk = "Mavi";
    public void setRenk(String renk){
        this.renk = renk;
    }
    public abstract double Alan();
    public abstract double Cevre();
}

public class Kare extends GeometrikSekil
{
    private double kenar;
    public Kare(double kenar){
        this.kenar = kenar;
    }

    @Override
    public double Cevre() {
        return 4*kenar;
    }

    @Override
    public double Alan() {
        return Math.pow(kenar, 2);
    }
}
```

6- Aşağıdaki özellikler Java programlama dilinde desteklenir mi? Destekleniyorsa örnek veriniz, desteklenmiyorsa nedenini belirtiniz? (10p)

- a) Operatör Overloading
- b) Pointer

Programlama Dillerinin Prensipleri Dersi 2020-2021 Bahar Dönemi Ara Sınavı

SINAV KURALLARI

Sınav Başlama Saati: 14 Nisan Çarşamba Günü Saat: 15:00

Sınav Bitiş ve SABİS'e Son Yükleme Saati: 14 Nisan Çarşamba Günü Saat: 15:50

- Cevap kağıdınızda, **Adınız, Soyadınız, Numaranız, Şubeniz ve İmzanız** mutlaka olmalıdır.
- Cevaplar kurşun kalem ile A4 kağıdına el yazısı ile yazılıp daha sonra taranıp SABİS'e yüklenmelidir.
- Herhangi bir soru cevabının kopya olması durumunda her iki tarafta ara sınav notundan sıfır alacaktır.
- Mail üzerinden kesinlikle gönderim kabul edilmemektedir.
- Hangi sorunun cevabının yazıldığı cevap kağıdında açıkça belirtilmelidir.

1-) Aşağıda verilen kodu öğrenci numaraniza göre oluşturunuz. Daha sonra oluşan kod üzerinde dinamik kapsam bağlamaya ve statik kapsam bağlamaya göre işlem yapıp ekran çıktılarını yazınız. Öğrenci numaranızın sondan bir önceki rakamı tek ise KOD1'i, çift ise KOD2'yi doldurunuz. 4. Bölüm için sondan bir önceki rakam dikkate alınacaktır. [25p]

[1. Bölüm] (Son rakam 0,1,2 olanlar)

```
cout<<x<<endl;
```

[1. Bölüm] (Son rakam 3,4,5 olanlar)

```
cout<<y<<endl;
```

[1. Bölüm] (Son rakam 6,7,8,9 olanlar)

```
if(x>y) cout<<x<<endl;
else cout<<y<<endl;
```

[2. Bölüm] (Son rakam 0,1 olanlar)

```
y=5;
```

[2. Bölüm] (Son rakam 2,3 olanlar)

```
y=4;
```

[2. Bölüm] (Son rakam 4,5 olanlar)

```
y=6;
```

[2. Bölüm] (Son rakam 6,7 olanlar)

```
y=7;
```

[2. Bölüm] (Son rakam 8,9 olanlar)

```
y=8;
```

[3. Bölüm] (Son rakam 0,1,2 olanlar)

```
cout<<x<<endl;
```

```
cout<<y<<endl;
```

[3. Bölüm] (Son rakam 3,4,5 olanlar)

```
cout<<y<<endl;
```

[3. Bölüm] (Son rakam 6,7,8,9 olanlar)

```
int x=25;
```

[4. Bölüm] (Sondan bir önceki rakam 0,1,2 olanlar)

```
CC(x);
```

[4. Bölüm] (Sondan bir önceki rakam 3,4,5 olanlar)

```
CC(y);
```

[4. Bölüm] (Sondan bir önceki rakam 6,7,8,9 olanlar)

```
CC(++x);
```

[1. Bölüm] (Son rakam 0,1,2 olanlar)

```
a=a/5;
```

[1. Bölüm] (Son rakam 3,4,5 olanlar)

```
a=b/5;
```

[1. Bölüm] (Son rakam 6,7,8,9 olanlar)

```
a=a/4;
```

[2. Bölüm] (Son rakam 0,1 olanlar)

```
int b=12;
```

[2. Bölüm] (Son rakam 2,3 olanlar)

```
int a=16;
```

[2. Bölüm] (Son rakam 4,5 olanlar)

```
a++;
```

[2. Bölüm] (Son rakam 6,7 olanlar)

```
b++
```

[2. Bölüm] (Son rakam 8,9 olanlar)

```
a=16;
```

[3. Bölüm] (Son rakam 0,1,2 olanlar)

```
cout<<XX()<<endl;
```

```
yazdir();
```

[3. Bölüm] (Son rakam 3,4,5 olanlar)

```
yazdir();
```

```
cout<<XX()<<endl;
```

[3. Bölüm] (Son rakam 6,7,8,9 olanlar)

```
int b=8;
```

```
yazdir();
```

```
cout<<XX()<<endl;
```

[4. Bölüm] (Sondan bir önceki rakam 0,1 olanlar)

```
int b=20;
```

[4. Bölüm] (Sondan bir önceki rakam 2,3 olanlar)

```
int a=20;
```

[4. Bölüm] (Sondan bir önceki rakam 4,5 olanlar)

```
int b=12;
```

[4. Bölüm] (Sondan bir önceki rakam 6,7 olanlar)

```
int a=12;
```

[4. Bölüm] (Sondan bir önceki rakam 8,9 olanlar)

```
int a=7;
```

```
// KOD1 Öğrenci numarasının Sondan bir önceki rakamı Tek ise
int x=100;
int y=10;

void yazdir(){
    [1. Bölüm]
}

void AA(){
    if(y<=0) return;
    cout<<y<<endl;
    [2. Bölüm]
    AA();
}

void CC(int y){
    [3. Bölüm]
    yazdir();
}

void BB(){
    int y=15;
    AA();
    [4. Bölüm]
}

int main(){
    int x=5;
    BB();
    CC(x);
    yazdir();
    return 0;
}
```

// KOD2 Öğrenci numarasının Sondan bir önceki rakamı Çift ise

```
int a=10;
```

```
int b=5;
```

```
int XX(){
```

```
    if(a<=1) return b;
```

```
    cout<<b<<endl;
```

```
    [1. Bölüm]
```

```
    return XX();
```

```
}
```

```
void yazdir(){
```

```
    [2. Bölüm]
```

```
    cout<<XX()<<endl;
```

```
}
```

```
void YY(){
```

```
    int a=25;
```

```
    [3. Bölüm]
```

```
}
```

```
void ZZ(){
```

```
    cout<<a<<endl;
```

```
    cout<<b<<endl;
```

```
}
```

```
int main(){
```

```
    [4. Bölüm]
```

```
    ZZ();
```

```
    YY();
```

```
    return 0;
```

```
}
```

2-) Aşağıda verilen Java sınıflarındaki A-B-C ve X-Y-Z bölümlerini A-B ve X-Y kısımlarını öğrenci numaranızın son rakamına göre **C ve Z kısımlarını öğrenci numaranızın sondan bir önceki rakamına göre doldurunuz. Doldurmadan önce öğrenci numaranızın sondan bir önceki rakamı tek ise Calisan sınıfını, çift ise Sekil sınıfını doldurunuz. Öğrenci numaranız ile ilişkili olmayan sınıf ve metodlar dikkate alınmamayacaktır. Daha sonra oluşan Java sınıfını son derste anlatıldığı gibi C dilinde **nesne yönelimli benzetimi** kullanarak tekrar yazınız. (**Zaman kaybetmemek için öğrenci numaranız ile ilgili olan metodu Java dilinde ilgili yere tekrar yazmadan direk dönüştürerek C dilinde yazınız.**) (Başlık ve Kaynak dosyası ayrı olacak şekilde yazılmacaktır.) [25p]**

| | |
|---|---|
| <p>A BÖLÜMÜ (Son rakam 0,1,2 olanlar)</p> <pre>public double IkramiyeHesapla(){ return (tecrubeYil*maas) / 2020; }</pre> <p>A BÖLÜMÜ (Son rakam 3,4,5 olanlar)</p> <pre>public void IkramiyeYazdir(){ System.out.println("ikramiye: "+ ((tecrubeYil*maas) / 2020)); }</pre> <p>A BÖLÜMÜ (Son rakam 6,7,8,9 olanlar)</p> <pre>public String Ikramiye(){ String ikramiye = ("ikramiye: "+ ((tecrubeYil*maas) / 2020)); return ikramiye; }</pre> | <p>X BÖLÜMÜ (Son rakam 0,1,2,3 olanlar)</p> <pre>public double AlanHesapla(){ return genislik*yukseklik; }</pre> <p>X BÖLÜMÜ (Son rakam 4,5,6 olanlar)</p> <pre>public void AlanYazdir(){ StringBuilder str = new StringBuilder(); str.append("Alan:"); str.append(genislik*yukseklik); System.out.println(str.toString()); }</pre> <p>X BÖLÜMÜ (Son rakam 7,8,9 olanlar)</p> <pre>public String AlanStr(){ StringBuilder str = new StringBuilder(); str.append("Alan:"); str.append(genislik*yukseklik); return str.toString(); }</pre> |
| <p>B BÖLÜMÜ (Son rakam 0,1,2 olanlar)</p> <pre>public double KatSayiHesapla(double[] donemselMaaslar){ double katSayi=1; outer: for(var maas : donemselMaaslar){ katSayi += maas/30; for(int i=0;i<7;i++){ if(katSayi%15 == 3) break outer; } } return katSayi; }</pre> <p>B BÖLÜMÜ (Son rakam 3,4,5 olanlar)</p> <pre>public double KatSayiHesapla(double[] donemselMaaslar){ double katSayi=1; for(var maas : donemselMaaslar){ katSayi += maas/30; } return katSayi; }</pre> <p>B BÖLÜMÜ (Son rakam 6,7,8,9 olanlar)</p> <pre>public String KatSayiHesapla(double maasOrtalaması){ Random rnd = new Random(); int ihtimal = rnd.nextInt(100); if(ihtimal<50) return "\u0000B"; else return "\u0000C"; }</pre> | <p>Y BÖLÜMÜ (Son rakam 0,1,2,3 olanlar)</p> <pre>public Sekil(double xy[][][]){ for(var s : xy) { for(var hucre : s) { genislik += hucre; yukseklik += hucre/2; } renklimi=false; } }</pre> <p>Y BÖLÜMÜ (Son rakam 4,5,6 olanlar)</p> <pre>public Sekil(double xy[][][]){ outer: for(var s : xy) { for(var hucre : s) { if(hucre < 0) break outer; genislik += hucre; yukseklik += hucre/2; } renklimi=false; } }</pre> |

| | |
|---|--|
| | <p>Y BÖLÜMÜ (Son rakam 7,8,9 olanlar)</p> <pre>public Sekil(String simbol){ switch(simbol) { case "\u000B": genislik = 50; yükseklik = 50; break; case "\u000C": genislik = 100; yükseklik = 100; break; } renklimi = true; }</pre> |
| <p>C BÖLÜMÜ (Sondan bir önceki rakam 0,1,2 olanlar)</p> <pre>public boolean mutlumu(){ Random rnd = new Random(); int r = rnd.nextInt(tecrubeYil); if(r > 5) return true; else return false; }</pre> <p>C BÖLÜMÜ (Sondan bir önceki rakam 3,4,5 olanlar)</p> <pre>public boolean mutlumu(String burc){ switch(burc) { case "BOGA": case "KOVA": return false; default: return true; } }</pre> <p>C BÖLÜMÜ (Sondan bir önceki rakam 6,7,8,9 olanlar)</p> <pre>public boolean mutlumu(String burc){ return burc == "KOVA" ? true : false; }</pre> | <p>Z BÖLÜMÜ (Sondan bir önceki rakam 0,1,2,3 olanlar)</p> <pre>public boolean temsilcimi(String simbol){ return simbol == "YAMUK" ? true : false; }</pre> <p>Z BÖLÜMÜ (Sondan bir önceki rakam 4,5,6 olanlar)</p> <pre>public boolean karemi(){ return genislik == yükseklik; }</pre> <p>Z BÖLÜMÜ (Sondan bir önceki rakam 7,8,9 olanlar)</p> <pre>public double[][][] pixel(){ Random rnd = new Random(); double[][][] noktalar = new double[10][10]; for(var s : noktalar) { for(int i=0;i<s.length;i++) { s[i] = rnd.nextInt(100); } } return noktalar; }</pre> |

| | |
|---|---|
| <p>//Öğrenci numarasının Sondan bir önceki rakamı Tek ise</p> <pre>class Calisan{ private double maas; private int tecrubeYil; boolean emeklimi; public Calisan(double maas,int tecrubeYil,boolean emeklimi) { this.maas = maas; this.tecrubeYil = tecrubeYil; this.emeklimi = emeklimi; } A BÖLÜMÜ B BÖLÜMÜ C BÖLÜMÜ }</pre> | <p>// Öğrenci numarasının Sondan bir önceki rakamı Çift ise</p> <pre>class Sekil{ private double genislik; private double yükseklik; boolean renklimi; public Sekil(double genislik,double yükseklik,boolean renklimi) { this.genislik = genislik; this.yükseklik = yükseklik; this.renklimi = renklimi; } X BÖLÜMÜ Y BÖLÜMÜ Z BÖLÜMÜ }</pre> |
|---|---|

3-) Aşağıdaki soruyu öğrenci numaranızın **sondan bir önceki rakamına göre cevaplayınız. Verilen ifadelerin gramerlerini yazınız. Öğrenci **numaranızın son rakamı** tek ise BNF, çift ise EBNF olarak yazılacaktır. [25p]**

| Sondan Bir Önceki Rakam | Grameri yazılacak ifade |
|-------------------------|---|
| 0 | a ile başlayıp b ile biten bütün string'leri kabul eder. Sadece a,b,c,d,e,f karakterleri dikkate alınacaktır. |
| 1 | a ile başlayıp ab ile biten bütün string'leri kabul eder. Sadece a,b,c,d,e,f karakterleri dikkate alınacaktır. |
| 2 | İçinde en az bir tane a olan bütün string'leri kabul eder. Sadece a,b,c,d,e,f karakterleri dikkate alınacaktır. |
| 3 | İçinde en az ab şeklinde bir string olan bütün string'leri kabul eder. Sadece a,b,c,d,e,f karakterleri dikkate alınacaktır. |
| 4 | ab ile başlayıp ba ile biten bütün stringleri kabul eder. Sadece a,b,c,d,e,f karakterleri dikkate alınacaktır. |
| 5 | aaa şeklinde en az bir string'in içinde bulunduğu bütün stringleri kabul eder. Sadece a,b,c,d,e,f karakterleri dikkate alınacaktır. |
| 6 | a,b,c,d,e,f ve bütün rakamların bulunduğu sözlük için rakam ile başlayan stringleri kabul eder. |
| 7 | a,b,c,d,e,f ve bütün rakamların bulunduğu sözlük için rakam ile başlamayan stringleri kabul eder. |
| 8 | a,b,c,d,e,f ve bütün rakamların bulunduğu sözlük için a ile başlamayan stringleri kabul eder. |
| 9 | a,b,c,d,e,f ve bütün rakamların bulunduğu sözlük için a ile başlayan ve rakam ile sonlanan stringleri kabul eder. |

4-) Aşağıdaki soruyu öğrenci numaranıza göre cevaplayınız. Öğrenci numaranızın sondan bir önceki rakam tek ise Bölüm 1, çift ise Bölüm 2'ye bakınız. Daha sonra öğrenci numaranızın son rakamına göre ilgili soruyu cevaplayınız? [25p]

| Bölüm 1 : Öğrenci numarasının Sondan bir önceki rakamı Tek ise | | Bölüm 2: Öğrenci numarasının Sondan bir önceki rakamı Çift ise | |
|--|--|--|---|
| Numara Son Rakamı | Cevaplayacağınız Soru | Numara Son Rakamı | Cevaplayacağınız Soru |
| 0 | Java programlama dili taşınabilirliği nasıl sağlamaktadır? | 0 | Derleyici ile yorumlayıcı arasındaki temel farkları belirtiniz. |
| 1 | Çoklu sınıf kalımının desteklendiği bir dil örneği veriniz. Ne gibi problem ile karşılaşabiliriz. | 1 | Kısa devre değerlendirme nedir? Açıklayınız ve desteklenmeyen bir dilde hataya sebep olabilecek kod örneğini yazınız. |
| 2 | Programlama dilleri değerlendirme kriterlerinden birbirine ters düşenleri yazıp nedenini açıklayınız. | 2 | Ara kod dönüştürme işleminin önemini belirtiniz. |
| 3 | Dilbilgisi belirsizliği ne demektir açıklayınız. Örnek veriniz. | 3 | Derlenme sürecinde makine bağımsız optimizasyon örneği veriniz. |
| 4 | Derlenme sürecinde makine bağımlı optimizasyon örneği veriniz. | 4 | İşlemci yükleme Java dilinde nasıl desteklenmektedir? Desteklenmiyorsa sebebi açıklayınız. |
| 5 | double ve float türlerinin bir arada kullanılmasının fayda ve zarar yönünden açıklayınız. Örnek veriniz. | 5 | Sallanan gösterici ne demektir? Örnek vererek açıklayınız. |
| 6 | Bellek bölgeleri düşünüldüğünde çöp olayı nasıl oluşmaktadır. | 6 | Dinamik tip bağlama nesne yönelimli dillerden nasıl gerçekleştirilebilir. Örnek vererek açıklayınız. |
| 7 | Belleğin 3 bölgesinde de sınıftan nesne oluşturulabilen dil hangisidir. Örnek vererek açıklayınız. | 7 | sizeof veya benzeri bir yapının Java dilinde olmamasının nedeni nedir? |
| 8 | Formal parametrenin üretilmediği parametre aktarma yöntemini açıklayınız ve örnek veriniz. | 8 | C dili başlık dosyaları ne işe yaramaktadır. Açıklayınız. |
| 9 | Komut satırı parametreleri ne işe yaramaktadır. C dilinde kod örneği veriniz. | 9 | Statik bellek bölgesinde C ve Java dillerinde değişken oluşturma kod örneğini yazınız. |