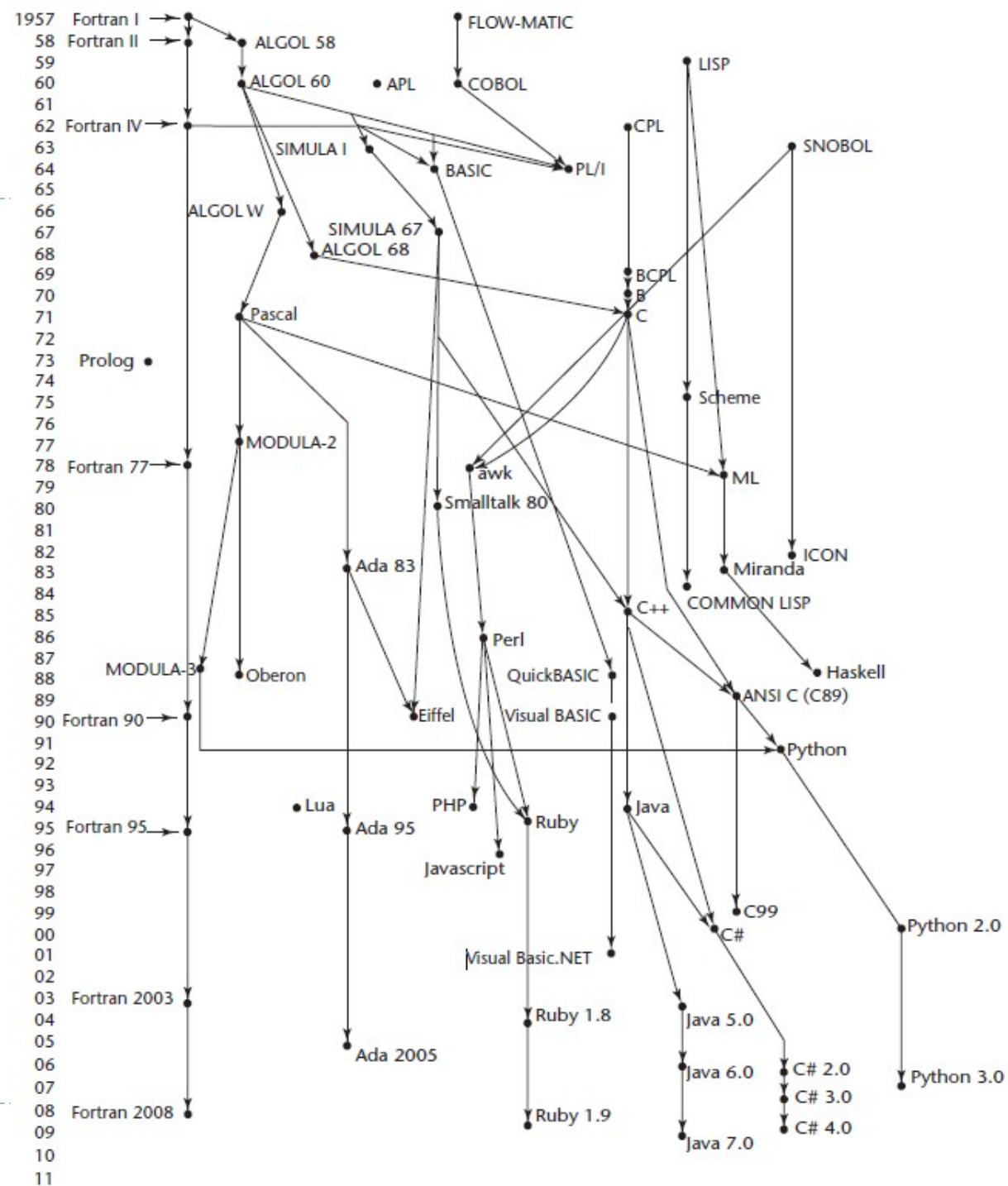


PROGRAMLAMA DİLLERİ

Hafta 5



Fortran

- ▶ **FOR**mul **TRAN**slating System
- ▶ 1954 yılında, IBM firmasında **John Backus** tarafından geliştirildi
- ▶ **İlk yüksek seviyeli** dil ve İlk ticari derlenen dil
- ▶ Aynı kişi 1959 yılında, yüksek seviyeli bir dilin yazış kurallarını açıklama noktasında standart bir notasyon haline gelen **Backus Naur Form** (BNF)'yi bulmuştur.
- ▶ Bilimsel Hesaplamalar, Mühendislik Uygulamaları ve Sayısal Analiz alanlarında yoğun olarak kullanılmaktadır



Fortran Örnek

C = "comment"

All-caps

For loop; I goes from 2 to 12
in increments of 1

Cols 1-6 for
comment
or stmt
label

```
C A PROGRAM TO COMPUTE MULTIPLICATION TABLES
  PROGRAM TABLES
    DO 20 I = 2,12
      PRINT *,I,' TIMES TABLE'
      DO 10 J = 1,12
        PRINT *,I,' TIMES' ,J,' IS' ,I*J
      CONTINUE
    END
```

Source: University of Strathclyde Computer Centre, Glasgow, Scotland

End of program

Fortran Örnek

```
// JOB
// FOR
*LIST SOURCE PROGRAM
*ONE WORD INTEGERS
C-----
C COMPUTE THE CRITICAL VALUES FOR A QUADRATIC EQN
C 0=A*X**2+B*X+C
C RETURNS DISCRIMINANT, ROOTS, VERTEX, FOCAL LENGTH, FOCAL POINT
C X1 AND X2 ARE THE ROOTS
C-----
      SUBROUTINE QUADR(A,B,C,DISCR,X1,X2,VX,VY,FL,FPY)
      REAL A,B,C,DISCR,X1,X2,VX,VY,FL,FPY

C DISCRIMINANT, VERTEX, FOCAL LENGTH, FOCAL POINT Y
      DISCR = B**2.0 - 4.0*A*C
      VX = -B / (2.0*A)
      VY = A*VX**2.0 + B*VX + C
      FL = 1.0 / (A * 4.0)
      FPY = VY + FL
      FL = ABS(FL)

C COMPUTE THE ROOTS BASED ON THE DISCRIMINANT
      IF(DISCR) 110,120,130

C -VE DISCRIMINANT, TWO COMPLEX ROOTS, REAL=X1, IMG=+/-X2
110   X1 = -B / (2.0*A)
      X2 = SQRT(-DISCR) / (2.0*A)
      RETURN

C ZERO DISCRIMINANT, ONE REAL ROOT
120   X1 = -B / (2.0*A)
      X2 = X1
      RETURN

C +VE DISCRIMINANT, TWO REAL ROOTS
130   X1 = (-B + SQRT(DISCR)) / (2.0*A)
      X2 = (-B - SQRT(DISCR)) / (2.0*A)
      RETURN

C
C NEXT STORE SUBROUTINE ON DISK USING DUP
      END
// DUP
```

```
// JOB
// FOR
*LIST SOURCE PROGRAM
*IOCS(CARD,1132 PRINTER)
*ONE WORD INTEGERS
C-----
C PROCESS DATA CARDS WITH A,B,C
C UNTIL A=0
C-----
      DATA ICARD,IPRT /2,3/
      REAL A,B,C
      REAL DISCR,XR1,XR2,VX,VY,FL,FPY

      WRITE(IPRT,901)
901   FORMAT(' -----')

C READ A B C, IF A=0 THEN EXIT
100   READ(ICARD,801)A,B,C
801   FORMAT(3F8.3)

C      EXIT WHEN A IS ZERO
      IF (A) 110,9000,110

C PRINT A B C
110   WRITE(IPRT,902)A,B,C
902   FORMAT(' QUADRATIC A=',F8.3,' B=',F8.3,' C=',F8.3)

C COMPUTE AND PRINT THE CRITICAL VALUES
      CALL QUADR(A,B,C,DISCR,XR1,XR2,VX,VY,FL,FPY)
      WRITE(IPRT,903) DISCR
903   FORMAT(' DISCRIMINANT=',F9.4)
      WRITE(IPRT,904) VX,VY
904   FORMAT(' VERTEX X=',F9.4,' Y=',F9.4)
      WRITE(IPRT,905) FL
905   FORMAT(' FOCAL LENGTH=',F9.4)
      WRITE(IPRT,906) VX,FPY
906   FORMAT(' FOCAL POINT X=',F9.4,' Y='F9.4)

      IF (DISCR) 120,130,140
```

LISP (1958)



- ▶ LISP : “**LIS**t **P**rocessing language”
- ▶ LISP yapay zeka uygulamaları için, MC Carthy ve ekibi tarafından 1958 yılında geliştirilmiştir.
- ▶ **Fonksiyonel** programlamayı destekler.
- ▶ Emacs ve AutoCAD
- ▶ Atomlar ve Listeler adı verilen iki veri yapısına sahiptir.
- ▶ Sayılar yerine sembolik ifadeler
- ▶ Program kodu ve veriler tam olarak aynı formdadır:
- ▶ İki LISP versiyonu Scheme (1970) ve COMMON LISP (1984)



LISP



- ▶ LISP esas olarak **yorumlayıcı** kullanan bir dildir. Derleyici kullanan versiyonları da vardır.
- ▶ Veri tiplmesi 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.
- ▶ **Çöp toplayıcı** desteği veren ilk dil
- ▶ Nesne Yönelimli Programlamayı destekler
- ▶ Çok iyi belgelendirilmiş olup, COMMON LISP, ANSI tarafından standart hale getirilmiştir.



LISP



```
;prefix notation
(+ 10 20)
;output 30
```

```
(* (+ 10 20) 3)
;output 30
```

```
;quoting
'(1 2 3)    abbrev (quote (1 2 3))
;output (1 2 3)
```

```
;list processing
(car '(1 2 3) )
; output 1
```

car & cdr were the original name of the IBM 704 assembler macros performing the operation

```
(cdr '(1 2 3))
; output (2 3)
```

```
(cons 1 '(2 3))
; output (1 2 3)
```

```
;function definition
(define factorial (n)
  (if (<= n 1)
      1
      (* n (factorial (- n 1)))
  )
)
```

```
(factorial 3)
; output 6
```

```
;metaprogramming
(cons '* (cdr '(+ 10 20)))
; output (* 10 20)
```

```
(eval
  (cons '* (cdr '(+ 10 20)))
)
; output 200
```

eval forces a further evaluation on the argument list

COBOL (1959)

- ▶ **CO**mmon **B**usiness **O**riented **L**anguage)
- ▶ 1959 yılında Grace Hopper ve Bob Bemer tarafından geliştirilmiştir.
- ▶ İngilizce sözcük yada cümle yapılarını kullanan ve İŞLETMELERe yönelik ortak bir dildir.
- ▶ bilgisayara büyük miktarda bilgi giriş-çıkışının yapıldığı uygulamalar için geliştirilmiştir.(Stok kontrol, Bordro)
- ▶ Katkıları
 - ▶ Hiyerarşik veri yapıları (records)
 - ▶ Uzun isim tanımlamalar
 - ▶ İç içe yapılar



COBOL

- ▶ COBOL içerisinde **rapor yazdırma**, **tablo içinde arama yapma**, ve kullanımı çok kolay olan **dosya sıralama rutinleri** bulunmaktadır.
- ▶ Yapısal özellikleri dolayısıyla veritabanı yönetim sistemleri ile birlikte kullanımı kolaydır.
- ▶ 2002'den bu yana OOP desteği vardır.
- ▶ Üzerinde hala çalışmalar yapılmakta ve Amerika hala büyük miktardaki verileri işlerken COBOL kullanmaktadır.

COBOL

Kaynak:

<https://www.designnews.com/design-hardware-software/cobol-coders-needed-coronavirus-fight>

The problem isn't just with unemployment systems. According to **Reuter**, COBOL code is used in almost 95 percent of ATM transactions. The language is even used to power 80 percent of in-person transactions. The report claims that there's still 220 billion lines of COBOL code currently being used in production today. Scarier still is that COBOL systems handle \$3 trillion in commerce every day.



COBOL Örnek

Program data
(variables)

```
$ SET SOURCEFORMAT "FREE"  
IDENTIFICATION DIVISION.  
PROGRAM-ID. Iteration-If.  
AUTHOR. Michael Coughlan.
```

Single-digit number

Initial value

```
DATA DIVISION.  
WORKING-STORAGE SECTION.  
01 Num1 PIC 9 VALUE ZEROS.  
01 Num2 PIC 9 VALUE ZEROS.  
01 Result PIC 99 VALUE ZEROS.  
01 Operator PIC X VALUE SPACE.
```

Source: <http://www.cs.tul.ie/COBOL/examples/conditn/IterIf.htm>

Level number; can be used
to express hierarchy (records)

Character

The source of Y2K bugs

ALGOL (1958)

- ▶ **ALGO**rithmic **L**anguage
- ▶ FORTRAN l'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.
- ▶ Makineden bağımsız ilk dil. Daha esnek ve daha güçlü bir dil olmuştur.
- ▶ Aşağıdaki gibi atama ifadesi ilk kullanım bu dilde olmuştur.

variable **:=** *expression*



ALGOL

- ▶ Blok yapısını öneren ilk dil (begin end)
- ▶ Kontrol yapılarına sahip ilk dil:
 - ▶ If then else
 - ▶ while
- ▶ Parametre aktarımı için yöntem öneren ilk dildir:
 - ▶ **pass by value**
 - ▶ **pass by name**
- ▶ **Rekürsiyon**'a izin verilmiştir.
- ▶ formal syntax (BNF) tanımlı ilk dildir. Bu formal diller, Parsing teorisi ve derleyici tasarımı konuları için bir başlangıçtır.
- ▶ Rekürsiyon ve blok yapısını desteklemek için donanımda yığıta izin verildiğinden bilgisayar mimarisini etkilemiştir.

ALGOL : Eksik yönler

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

ALGOL

comment ALGOL 60 Example Program

Input: An integer, listlen, where listlen is less than 100, followed by listlen-integer values

Output: The number of input values that are greater than the average of all the input values ;

begin

integer array intlist [1:99];

integer listlen, counter, sum, average, result;

sum := 0;

result := 0;

readint (listlen);

if (listlen > 0) ^ (listlen < 100) then

begin

comment Read input into an array and compute the average;

for counter := 1 step 1 until listlen do

begin

readint (intlist[counter]);

sum := sum + intlist[counter]

end;

comment Compute the average;

average := sum / listlen;

comment Count the input values that are > average;

for counter := 1 step 1 until listlen do

if intlist[counter] > average

then result := result + 1;

comment Print result;

printstring("The number of values > average is:");

printint (result)

end

else

printstring ("Error-input list length is not legal");

end



BASIC (1964)

- ▶ Beginner's All-purpose Symbolic Instruction Code (1964)
- ▶ John Kemeny ve Thomas Kurtz
- ▶ Öğrencilerin bilgisayara daha kolay erişimlerini sağlamak ve **basit** ve etkin bir programlama dili ile program yazabilme isteklerine cevap vermek için tasarlanmış bir dildir.
- ▶ Sadece 4 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"

BASIC

- ▶ **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 pek çok versiyonları olmuştur. 1989'da ise nesne yönelimli uyarlama olan Visual BASIC ve 1998'de VB6.0 sunulmuştur.

Example (Applesoft BASIC)

Variables

Lines
numbered

```
10 INPUT "What is your name: "; U$
20 PRINT "Hello "; U$
25 REM
30 INPUT "How many stars do you want: "; N
35 S$ = ""
40 FOR I = 1 TO N
50 S$ = S$ + "*"
55 NEXT I
60 PRINT S$
65 REM
70 INPUT "Do you want more stars? "; A$
80 IF LEN(A$) = 0 THEN GOTO 70
90 A$ = LEFT$(A$, 1)
100 IF (A$ = "Y") OR (A$ = "y") THEN GOTO 30
110 PRINT "Goodbye ";
120 FOR I = 1 TO 200
130 PRINT U$; " ";
140 NEXT I
150 PRINT
```

Comment

Loop

Goto
common

Pascal (1971)

- ▶ 1971 yılında Niklaus Wirth tarafından Zurih'te geliştirildi.
- ▶ Algol dilini daha basit ve sadeleştirilmiş halidir.
- ▶ Eğitim amaçlı kullanım
- ▶ Basit ve okunabilir, yazılabilir (expressive)
- ▶ Fortran ve C ile karşılaştırıldığında emniyetli bir dildir.
- ▶ OOP destekli hali Delphi 1995 yılında Anders Hejlsberg tarafından geliştirildi.



Pascal Örnek

```
var S, T : set of 1..10;  
S := [1, 2, 3, 5, 7];  
T := [1..6];  
U := S + T;    { set union }  
if 6 in S * T then ... { set intersection }  
– (Note comments in {}'s)
```



Pascal Örnek

```
program mine(output);  
  var i : integer;  
  
  procedure print(var j: integer);  
  
    function next(k: integer): integer;  
    begin  
      next := k + 1  
    end;  
  
    begin  
      writeln('The total is: ', j);  
      j := next(j)  
    end;  
  
  begin  
    i := 1;  
    while i <= 10 do print(i)  
  end.
```



SIMULA 67

- ▶ Ole-Johan Dahl and Kristen Nygaard tarafından 1967 yılında geliştirilmiştir.
- ▶ İlk olarak simülasyon için tasarlanmış bir dildir. Daha sonra genel amaçlı hale geldi.
- ▶ ALGOL 60'ın genişletilmiş versiyonudur.
- ▶ Daha önce durdurulduğu yerden itibaren yeniden çalışmaya başlayan altprogramları desteklemektedir.
- ▶ Veri soyutlamasına imkan veren **sınıf yapılarını** destekleyen ilk dil. Nesneye dayalı mantığın ilk adımları

SIMULA 67

- ▶ Öre çıkan özellikleri:
 - ▶ Class ve objectler
 - ▶ Miras alma
 - ▶ Nesnelere pointer
 - ▶ Call by reference
 - ▶ Garbage collection
 - ▶ concurrency

Example

Parameter types

```
class Point(x,y); real x,y;
```

```
begin
```

```
  boolean procedure equals(p); ref(Point) p;
```

null pointer

```
  if p != none then
```

Field access

```
    equals := abs(x - p.x) + abs(y - p.y) < 0.00001
```

```
  real procedure distance(p); ref(Point) p;
```

```
    if p == none then error else
```

```
      distance := sqrt((x - p.x)**2 + (y - p.y)**2);
```

```
end ***Point***
```

```
p := new Point(1.0, 2.5);
```

```
q := new Point(2.0, 3.5);
```

```
if p.distance(q) > 2 then ...
```

Pointer assignment

Source: <http://www.stanford.edu/class/cs242/slides/2004/simula-smalltalk.pdf>

Prolog

- ▶ **PRO**gramming **LOG**ic
- ▶ 1972 yılında Alain Colmerauer tarafından geliştirilmiştir.
- ▶ İlk **mantık paradigmasını** kullanan dili
- ▶ Mantık yürütme ve ispatlama tekniklerini kullanır
- ▶ PROLOG programı, bir nesneler kümesi ile bu nesnelerle ilişkili hedeflere nasıl erişilebileceğini tanımlayan kurallar kümesinden oluşur.
- ▶ Prosedürel bir yaklaşım değildir.
- ▶ Dil kurallar ve gerçeklerden oluşur.
- ▶ Verimsiz kalmıştır.
- ▶ Kullanım alanı yapay zeka alanlarıyla sınırlıdır.

Prolog

```
eval(v(X)) :- true(v(X)).
```

```
eval(and(A,B)) :-  
    eval(A), eval(B).
```

```
eval(or(A,B)) :- eval(A).  
eval(or(A,B)) :- eval(B).
```

```
eval(not(A)) :- not eval(A).
```

```
true(v(p1)).  
true(v(p3)).
```

```
//eval: (p1 and p2) or p3  
?- eval(or(and(v(p1),v(p2)),v(p3))).  
true.  
?- eval(v(p3)).  
false.
```

```
sat(A) :- eval(A).
```

```
true(v(p1)).           //true.  
true(v(p1)) :- fail. //false
```

```
true(v(p2)).           //true.  
true(v(p2)) :- fail. //false
```

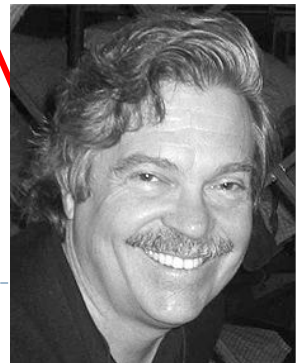
```
true(v(p3)).           //true.  
true(v(p3)) :- fail. //false
```

```
//sat: ((p1 and p2) and not  
        p2)  
?- sat(and(and(v(p1),v(p2)),  
          not(v(p1)))).  
false.
```



Smalltalk (1970'ler)

- ▶ Xerox PARC, Alan Kay tarafından geliştirilmiştir.
- ▶ OOP 3 temel karakteristiği: data abstraction, inheritance ve dynamic binding
- ▶ Smalltalk sadece bir PL değil, aynı zamanda yazılım geliştirme aracıdır. **IDE desteği.**
- ▶ Program birimleri, verileri kapsülleyen nesneler ve yöntemlerden oluşur.
- ▶ Hesaplama bir nesneye bir mesaj göndererek yapılır
- ▶ **Tasarım şablonlarını ilk kullanan dil.** Öne çıkan **MV**



Smalltalk

- ▶ İlk olarak Smalltalk-80 versiyonu Xerox dışında ticari amaçlı yazılımlarda kullanıldı. Halen yazılım geliştirmeleri için yeni teknolojileri de içererek kullanılmaktadır.
- ▶ Smalltalk tamamen nesne yönelimli olan ve ticari ilk programlama dilidir.
- ▶ Kaynak kodunun açık olması iyi bir **eğitim** ortamı haline gelmesini sağlamıştır. Diğer dillere göre çok basit bir sentaksı vardır ve Java gibi Geniş ve sürekli genişleyen bir programlama kütüphanesine sahiptir. Platform bağımsız bir dildir.
- ▶ Bu dilde nesne yönelimli programlamanın 3 temel karakteristiği olan veri soyutlama (data abstraction), kalıtım (inheritance) ve dinamik bağlama(dynamic binding) kavramlarının hepsi bulunmaktadır.

Smalltalk

command.central

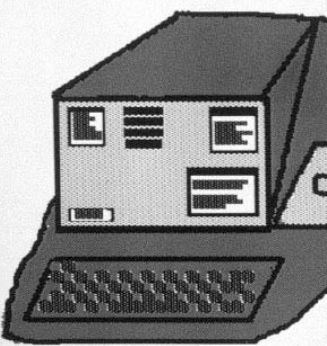
```
quit then 'dmt.boot.'
notes mids ev @erase.
usersetup.ft usereditor.ft
usertextwindow.ft userfileseg.ft
window.ft format.ft diana.ft
blits.ft commander.ft tiles.ft
filin 'press.ft'. dsofl. @pl*pressfile
'jumbo.press'. pl bitmap
STIDESPLAY rectangle. pl close.
dson.
sched + figwindow 'coreuse'
sch
```

diana.ft.
userfont.ft.
usersetup.ft.
userpics.ft.
format.ft.
userpictools.ft.

notes.

```
avoid full show for esc,
cut, paste, (...)
auto-clean up
get Larry's views
Clean windows
para, file, font
fig, pic, dock
debugger, view
write up
boolean expus
code obj
bootstrap
read run, files
new/init
Clean's mods
scheduling - interrupts
```

notetaker.pic



Leading

XEROX
PALO ALTO RESEARCH CENTER
Learning Research Group
October 29, 1978

undo
copy
paste
cut
do it!
(...)
open
include

text

To: SMALLTALK Interest

From: Dan Ingalls as scribe

Subject: Message syntax

Filed on: CINGALLS>XTinterpreter.Bravo

This is a working paper describing the next SMALLTALK system. It is the result of many proposals and discussions among L&G and friends.

clock

Thursday
February 3, 1977
11:11am

Interpretation

The operation of the bytecode interpreter is very simple. The code consists of sequences such as d₁ d₂ d₃ s₁, or three data bytes followed by a selector byte. Each of the data bytes in turn gets resolved to a full (16-bit) object pointer and pushed onto the stack.

System Browser

Collections-Sequence	Interval	accessing	collect:
Collections-Text	LinkedList	copying	do:
Collections-Array	MappedCollection	adding	do:andBetweenDo:
Collections-Stream	OrderedCollection	removing	promoteFirstSuchT
Collections-Support	SortedCollection	enumerating	reverse
Graphics-Primitives		private	reverseDo:
Graphics-Display C			select: Form Editor
Graphics-Media			
Graphics-Paths			
instance	class		

collect: aBlock

"Evaluate aBlock with each of my elements as the argument. C...
resulting values into a collection that is like me. Answer with
collection. Override superclass in order to use add, not at:put:.

```
| newCollection |
newCollection + self species new.
self do: [:each | newCollection add: (aBlock value: each)].
+newCollection
```

User Interrupt

```
Paragraph>>characterBlockAtPoint:
Paragraph>>mouseSelectto:
CodeController(ParagraphEditor)>>processRedButton
CodeController(ParagraphEditor)>>processMouseButton
CodeController(ParagraphEditor)>>controlActivity
CodeController(Controller)>>controlLoop
```

controlActivity

```
self scrollBarContainsCursor
ifTrue:
    [self scroll]
ifFalse:
    [self processKeybo
    self processMouse
```

blueButton
scrollBar
marker
savedAre
paragraph
startBloc

31@537 corner:
63@770

[<Robson>SF]*

[File]><Robson>SF>ScreenForm.st

[File]><Robson>SF>ScreenFormChanges.st

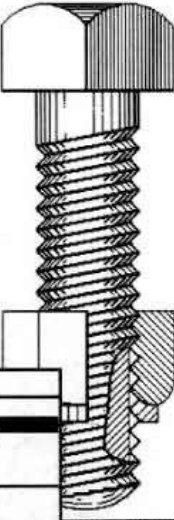
[File]><Robson>SF>WordGraphics.form

Rectangle fromUser origin

ScreenForm setFullPageWidth.

(Form readFrom: 'FilledSkate.form') edit

Fig. 1.



Smalltalk



C (1972)

- ▶ 1972 yılında Dennis Ritchie (Bell labs) tarafından geliştirilmiştir.
- ▶ UNIX işletim sistemi geliştirilmesi için
- ▶ Assemble yakın performans
- ▶ Kütüphane ile I/O ve dinamik bellek yönetimi
- ▶ Pointers



C



```
#include <stdio.h>
```

```
int main()
{
    int nums[5] = {20, 31, 17, 47, 14};
    int nums_length = 5;

    printf("Unsorted Array:\n");
    display_array(nums, nums_length);

    bubble_sort(nums, nums_length);

    printf("Sorted Array:\n");
    display_array(nums, nums_length);
    return 0;
}
```

```
/* Function that simply displays each element of
input_array. */
void display_array(int input_array[], int
input_size)
{
    int i;

    for (i = 0; i < input_size; i++)
    {
        printf("%d ", input_array[i]);
    }
    printf("\n\n");
}
```

```
/* Bubble sort function. Sorts an array of ints
in descending order. */
void bubble_sort(int array[], int arrayLength)
{
    int i, j, flag = 1;
    int temp;

    for(i = 1; (i <= arrayLength) && flag; i++)
    {
        flag = 0;
        for(j = 0; j < (arrayLength - i); j++)
        {
            if (array[j + 1] > array[j])
            {
                temp = array[j + 1];
                array[j + 1] = array[j];
                array[j] = temp;
                flag = 1;
            }
        }
    }
}
```

ADA

- ▶ ABD savunma bakanlığının bir çalışması sonucu ortaya çıkmıştır. Gömülü sistemlerin programlanması için geliştirilmiştir (1983).
- ▶ Augusta Ada Byron'a itafen isimlendirilmiştir.
- ▶ Blok yapılı, nesne yönelimli, genel amaçlı ve eşzamanlılığı destekleyen bir dildir.
- ▶ Büyük boyutlu yazılımlar için uygundur.



Ada

Ada example (definition)

```
package STACKPACK is
  type STACKTYPE is limited private;
  function EMPTY (S : in STACKTYPE) return BOOLEAN;
  procedure PUSH (S : in out STACKTYPE;
                  ELEMENT : in INTEGER);
  procedure POP (S : in out STACKTYPE);
  function TOP (S : in STACKTYPE) return INTEGER;
```

```
private
  type LIST_TYPE is array (1..100) of INTEGER;
  type STACKTYPE is
    record
      LIST : LIST_TYPE;
      TOPSUB : INTEGER range 0..100 := 0;
    end record;
end STACKPACK;
```

```
with TEXT_IO; use TEXT_IO;
package body STACKPACK is
  function EMPTY (S : in STACKTYPE) return BOOLEAN is
  begin
    if S.TOPSUB = 0
    then return TRUE;
    else return FALSE;
    end if;
  end EMPTY;

  procedure PUSH (S : in out STACKTYPE;
                  ELEMENT : in INTEGER) is
  begin
    if S.TOPSUB >= 100
    then
      PUT_LINE ("ERROR - Stack overflow");
    else
      S.TOPSUB := S.TOPSUB + 1;
      S.LIST(TOPSUB) := ELEMENT;
    end if;
  end PUSH;

  procedure POP (S : in out STACKTYPE) is
  begin
    if S.TOPSUB = 0
    then PUT_LINE ("ERROR - Stack underflow");
    else S.TOPSUB := S.TOPSUB - 1;
    end if;
  end POP;

  function TOP (S : in STACKTYPE) return INTEGER is
  begin
    if S.TOPSUB = 0
    then PUT_LINE ("ERROR - Stack underflow");
    else return S.LIST(S.TOPSUB);
    end if;
  end TOP;
end STACKPACK;
```

C++ (1983)


- ▶ 1983 Bell Labs, Bjarne Stroustrup
- ▶ C ve Simula
- ▶ Çoklu miras, template, metot overloading
- ▶ Fonksiyon ve NYP 'ya izin verilmektedir.
- ▶ Kuvvetli tip ayrımı, **dinamik bellek yönetim** şablonlara sahip olma ve çok biçimlilik (polymorphism) özellikleri vardır.
- ▶ C++ kütüphanesi
 - ▶ Containers, algorithms, iterators
 - ▶ Template



C++ Örnek

```
#include <iostream>
int main ()
{
    int n;
    cout << "Enter number > ";
    cin >> n;
    while (n>0) {
        cout << n << ", ";
        --n;
    }
    cout << "Done!\n";
    return 0;
}
```

Overloaded operators



C++ Örnek

```
// File Insieme-SS.h
#ifndef INSIEME_SS_H
#define INSIEME_SS_H

struct nodo {
    int info;
    nodo* next;
};

class Insieme
{public:
    Insieme();
    Insieme(const Insieme&);
    ~Insieme();
    Insieme& operator=(const Insieme&);
    bool EstVuoto();
    bool Membro(int);
    void Inserisci(int);
    void Elimina(int);
    int Scegli();
private:
    nodo* inizio;
    static bool Appartiene(int,nodo*);
    static void Cancella(int,nodo*&);
    static nodo* Copia(nodo*);
    static void Rilascia(nodo*);
};
#endif
```

```
// File Insieme-SS.cpp
#include "Insieme-SS.h"

Insieme::Insieme()
{ inizio = NULL; }

Insieme::Insieme(const Insieme& t)
{ inizio = Copia(t.inizio); }

Insieme::~Insieme()
{ Rilascia(inizio); }

Insieme& Insieme::operator=(const Insieme& t)
{ if (this != &t)
    { Rilascia(inizio);
      inizio = Copia(t.inizio);
    }
  return *this;
}
...
```

Java (1995)

- ▶ 1995 yılında **James Gosling** (Sun Microsystem) tarafından geliştirilmiştir.
- ▶ Java, basit, taşınabilir ve nesneye yönelik özellikte bir dildir.
- ▶ Miras alma, çok biçimlilik, kuvvetli tip kontrolü, eş zamanlılık kontrolü, dinamik olarak yüklenebilen kütüphaneler, diziler, string işlemleri ve standart kütüphane gibi özellikleri vardır.
- ▶ Bir Java programının temel yapısal bileşeni sınıftır. Bütün veri ve metotlar bir sınıf ile ilişkilidir. Global veri yada fonksiyon yoktur.
- ▶ Hem referans değişkenleri ile hem de ilkel tiplerle erişilebilen sınıflara sahiptir.
- ▶ C++'ta bulunan çoklu miras alma, operatörlerin üst üste bindirilmesi, ve makro ön işlemcisi özellikleri Java'da yoktur.
- ▶ Java'da şablon yapıları yoktur.

Java

- ▶ Pointer yoktur.
- ▶ Records, union or enumeration tipler yoktur.
- ▶ Prosedürel programlamayı desteklemez.
- ▶ Sadece tekli miras almayı destekler
- ▶ İplik (**threads**) yapısına sahip olduğundan eşzamanlılığı yönetmek kolaydır.
- ▶ **Çöp toplama (Garbage collection)** nesneler için belleği eniyi kullanımı sağlar.
- ▶ Tip dönüşümü kuvvetlidir.
- ▶ VM sayesinde güvenlik katmanı sağlanır.

Java

- ▶ Java tipik olarak platformdan bağımsız olarak byte kodları biçiminde derlenir. Daha sonra bu byte kodlar bir Java sanal makinesi adı verilen bir Java yorumlayıcısı tarafından kullanılacağı platformun makine dilindeki koduna çevrilir. Derlenmiş Java sınıflarının taşınabilirliğini garantileyen bir özellik vardır. Oda .class format adı verilen Java byte-kod dosya formatının kesin olarak tanımlanmış olmasıdır.
- ▶ Java internet, web ve mobil programlamada yaygın olarak kullanılmaktadır.

C# (2001)

- ▶ 2001, Microsoft, Anders Hejlsberg
- ▶ Imperative, OOP
- ▶ Javadan farkları
 - ▶ Pointers
 - ▶ Kullanıcı tanımlı türler (C structs)
 - ▶ SQL benzeri dahili sorgular



C#

```
using System;
class Employee { }
class Contractor : Employee { }
class CastExample
{
    public static void Main ()
    {
        Employee e = new Employee();
        Console.WriteLine("e = {0}",
            e == null ? "null" : e.ToString());
        Contractor c = e as Contractor ;
        Console.WriteLine("c = {0}",
            c == null ? "null" : c.ToString());
    }
}
```



Swift (2014)

- ▶ Apple, Chris Lattner
- ▶ Rust, Haskell, Ruby, Python, C# gibi dillerden etkilenme
- ▶ Ön tanımlı pointer üretilememesi
- ▶ Java benzeri metot çağırma yöntemi



Swift

```
func str2i(s:String) -> int
{
    o->f(i)
}
```

```
func int2s(i:int) -> String
{
```

```
    var str = "hello"
    str += " world"
```

```
    let int: x = 1
```

```
    var y = 2
```

```
}
```

Method return type

Parameter type

Method invocation

Constant value

Explicit type

Variable value

Type inference



Popüler diller (tiobe)

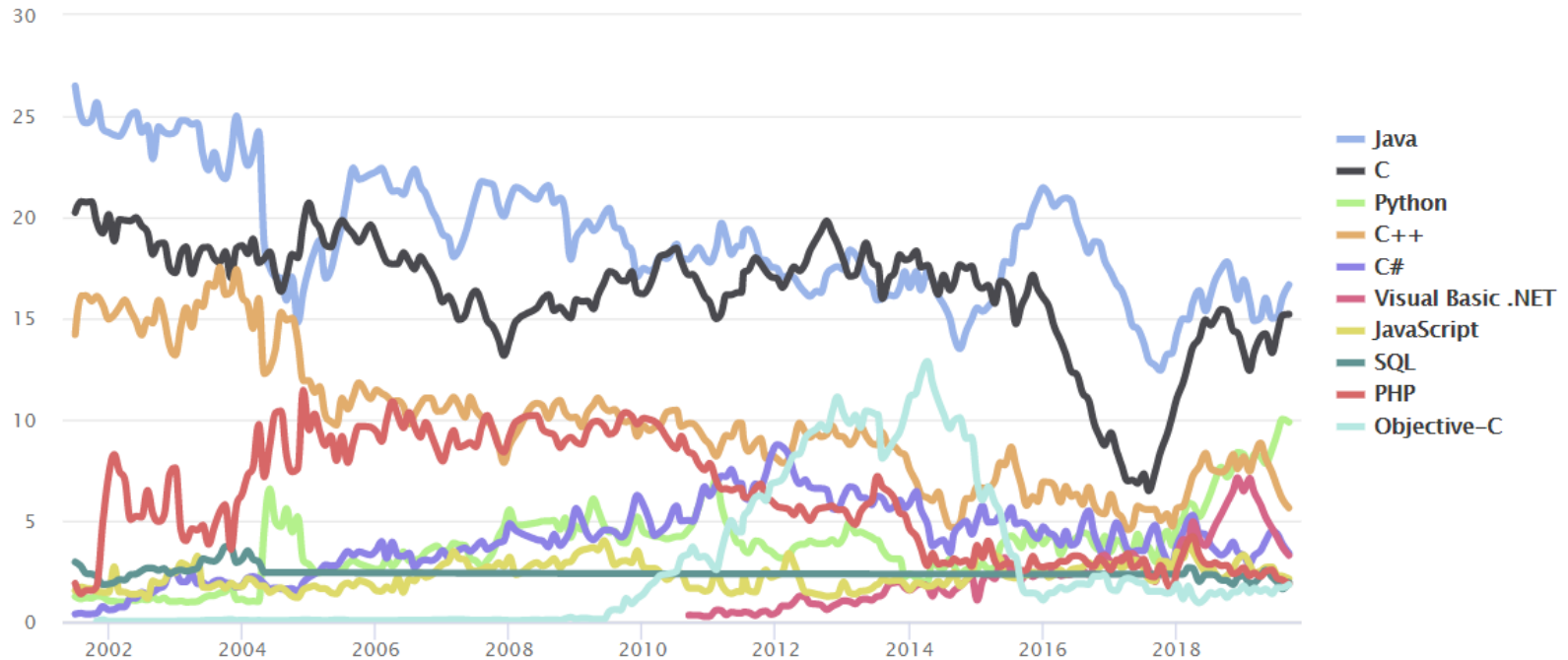
Sep 2019	Sep 2018	Change	Programming Language	Ratings	Change
1	1		Java	16.661%	-0.78%
2	2		C	15.205%	-0.24%
3	3		Python	9.874%	+2.22%
4	4		C++	5.635%	-1.76%
5	6	⬆	C#	3.399%	+0.10%
6	5	⬇	Visual Basic .NET	3.291%	-2.02%
7	8	⬆	JavaScript	2.128%	-0.00%
8	9	⬆	SQL	1.944%	-0.12%
9	7	⬇	PHP	1.863%	-0.91%
10	10		Objective-C	1.840%	+0.33%
11	34	⬆	Groovy	1.502%	+1.20%
12	14	⬆	Assembly language	1.378%	+0.15%
13	11	⬇	Delphi/Object Pascal	1.335%	+0.04%
14	16	⬆	Go	1.220%	+0.14%
15	12	⬇	Ruby	1.211%	-0.08%
16	15	⬇	Swift	1.100%	-0.12%



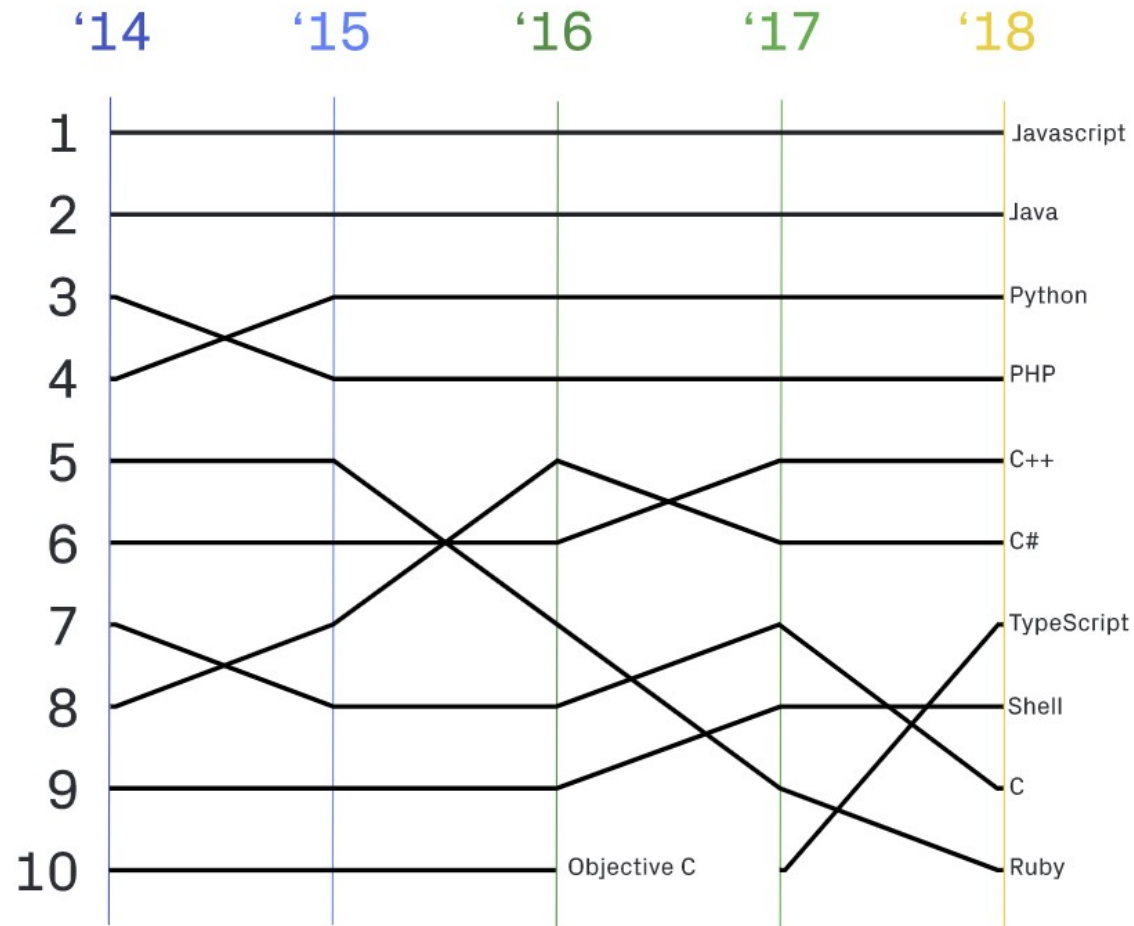
Popüler diller (tiobe)

TIOBE Programming Community Index

Source: www.tiobe.com



Popüler diller (github)



En hızlı büyüyen diller (github)

Growth in contributors

1	Kotlin	2.6x
2	HCL	2.2x
3	TypeScript	1.9x
4	PowerShell	1.7x
5	Rust	1.7x
6	CMake	1.6x
7	Go	1.5x
8	Python	1.5x
9	Groovy	1.4x
10	SQLPL	1.4x



Bazı dillerin özellikleri

Language	Paradigm	Type System	Type Checking	Variable Declarations	Type Declarations
Java	OO/Imp	Strong	Static	Explicit	Explicit
C	Imperative	Weak	Static	Explicit	Explicit
Ruby	OO/Scripting	Strong	Dynamic	Implicit	None
OCaml	Functional	Strong	Static	Explicit	Inferred
Rust	Imp/Func	Strong	Static	Explicit	Expl/Infer



Kaynaklar

- ▶ Programlama Dillerinin Prensipleri, Prof. Dr. Nejat Yumuşak, Dr. Muhammed Fatih Adak, Seçkin Yayıncılık
- ▶ Sakarya Üniversitesi, Bilgisayar ve Bilişim Mühendisliği Programlama dilleri ve kavramları Ders Sunumları
- ▶ Concepts of Programming Languages (11. ed.), Robert W. Sebesta sunumları
- ▶ Dr. Erkan Duman, Programlama Dilleri ders notları
- ▶ Harran Üniversitesi, Programlama dilleri ders sunumları

