Closures:

The next great development
in programming technology

Jules May

julesmay.co.uk

AGC

```
# ENTRY IS THROUGH 1/ACCJOB OR 1/ACCSIT WHEN 1/ACCS IS TO BE DONE AS A SEPARATE NOVAC JOB.
# IT IS POSSIBLE FOR MORE THAN ONE OF THESE JOBS TO BE SET UP CONCURRENTLY.  HOWEVER, SINCE THERE IS NO CHECK OF
# NEWJOB, A SECOND MANIFESTATION CANNOT BE STARTED UNTIL THE FIRST IS COMPLETED.

1/ACCSET  CAF  ZERO      # ENTRY FROM FRESH START/RESTART CODING.
          TS   AOSQ      #    NULL THE OFFSET ESTIMATES FOR 1/ACCS.
          TS   AOSR
          TS   ALPHAQ        #    NULL THE OFFSET ESTIMATES FOR DOWNLIST
          TS   ALPHAR

1/ACCJOB  TC   BANKCALL  # 1/ACCS ASSUMES ENTRY VIA BANKCALL.
          CADR 1/ACCS    +2   # SKIP EBANK SETTING.
          TC   ENDOFJOB

1/ACCS    CA   EBANK6        # ***** EBANK SET BUT NOT RESTORED *****
          TS   EBANK

          TC   MAKECADR  # SAVE RETURN SO THAT BUF2 MAY BE USED
          TS   ACCRETRN

# DETERMINE MASS OF THE LEM.

          CA   DAPBOOLS  # IS THE CSM DOCKED
          MASK CSMDOCKD
          TS   DOCKTEMP  # STORE RECORD OF STATE IN TEMP (MPAC +3).
          CCS  A
          CS   CSMMASS       #    DOCKED:  LEMMAS = MASS - CSMMASS
          AD   MASS      #    LEM ALONE:  LEMMASS = MASS
          TS   LEMMASS
```

FORTRAN IV

```
C AREA OF A TRIANGLE - HERON'S FORMULA
C INPUT - CARD READER UNIT 5, INTEGER INPUT, ONE BLANK CARD FOR END-OF-DATA
C OUTPUT - LINE PRINTER UNIT 6, REAL OUTPUT
C INPUT ERROR DISPAY ERROR MESSAGE ON OUTPUT
  501 FORMAT(3I5)
  601 FORMAT(4H A= ,I5,5H  B= ,I5,5H  C= ,I5,8H  AREA= ,F10.2,
     $13H SQUARE UNITS)
  602 FORMAT(10HNORMAL END)
  603 FORMAT(23HINPUT ERROR, ZERO VALUE)
      INTEGER A,B,C
   10 READ(5,501) A,B,C
      IF(A.EQ.0 .AND. B.EQ.0 .AND. C.EQ.0) GO TO 50
      IF(A.EQ.0 .OR.  B.EQ.0 .OR.  C.EQ.0) GO TO 90
      S = (A + B + C) / 2.0
      AREA = SQRT( S * (S - A) * (S - B) * (S - C) )
      WRITE(6,601) A,B,C,AREA
      GO TO 10
   50 WRITE(6,602)
      STOP
   90 WRITE(6,603)
      STOP
      END
```

FORTRAN 77

```
*     euclid.f (FORTRAN 77)
*     Find greatest common divisor using the Euclidean algorithm

      PROGRAM EUCLID
        PRINT *, 'A?'
        READ *, NA
        IF (NA.LE.0) THEN
          PRINT *, 'A must be a positive integer.'
          STOP
        END IF
        PRINT *, 'B?'
        READ *, NB
        IF (NB.LE.0) THEN
          PRINT *, 'B must be a positive integer.'
          STOP
        END IF
        PRINT *, 'The GCD of', NA, ' and', NB, ' is', NGCD(NA, NB), '.'
        STOP
      END

      FUNCTION NGCD(NA, NB)
        IA = NA
        IB = NB
    1   IF (IB.NE.0) THEN
          ITEMP = IA
          IA = IB
          IB = MOD(ITEMP, IB)
          GOTO 1
        END IF
        NGCD = IA
        RETURN
      END
```

Pascal

```
PROGRAM Sort(input, output);
    CONST
        MaxElts = 50;
    TYPE
        IntArrType = ARRAY [1..MaxElts] OF Integer;
    VAR
    i, j, tmp, size: integer;
    arr: IntArrType;

        PROCEDURE ReadArr(VAR size: Integer; VAR a: IntArrType);
        BEGIN
            size := 1;
            WHILE NOT eof DO BEGIN
                readln(a[size]);
                IF NOT eof THEN
                    size := size + 1
            END
        END;

BEGIN
    ReadArr(size, arr);

    (* Sort using bubble sort. *)
    FOR i := size - 1 DOWNTO 1 DO
        FOR j := 1 TO i DO
            IF arr[j] > arr[j + 1] THEN BEGIN
                tmp := arr[j];
                arr[j] := arr[j + 1];
                arr[j + 1] := tmp;
            END;

    FOR i := 1 TO size DO
        writeln(arr[i])
END.
```

C

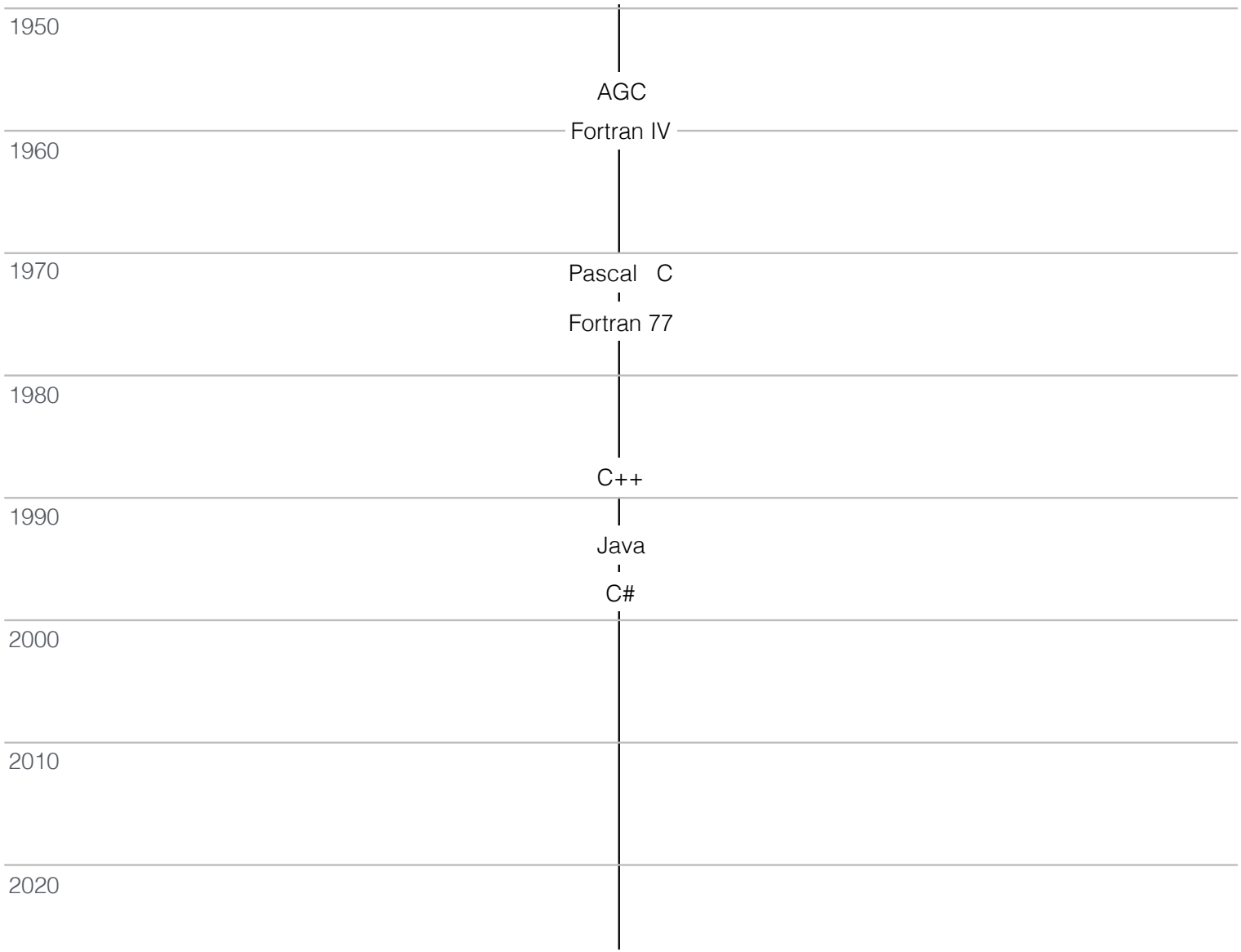
```c
#include<stdio.h>
#include<conio.h>

int fact(int);

int main() {
   int factorial, num;

   printf("Enter the value of num :");
   scanf("%d", &num);

   factorial = fact(num);
   printf("Factorial is %d", factorial);

   return (0);
}

int fact(int n) {
   if (n == 0) {
      return (1);
   }
   return (n * fact(n - 1));
}
```

C++

```
#include <iostream>
using namespace std;

class Date {
  private:
    int year;
    int month;
    int day;

  public:
    Date (int d, int m, int y) {
        if(d>0 && d<31) day = d;
        if(m>0 && m<13) month = m;
        if(y>0) year =y;
    }

    void print() {
        cout << day << "-" << month << "-" << year << endl;
    }
};

int main() {
   Date today(1,9,1999);

   cout << "This program was written on ";
   today.print();
   return 0;
}
```

Java

```
public class CallingMethodsInSameClass
{
    public static void printOne() {
        System.out.println("Hello World");
    }

    public static void printTwo() {
        printOne();
        printOne();
    }

    public static void main(String[] args) {
        printOne();
        printOne();
        printTwo();
    }
}
```


1950
AGC
Fortran IV
1960
1970
Pascal
C
Fortran 77
1980
C++
1990
Java
C#
2000
2010
2020

Algol

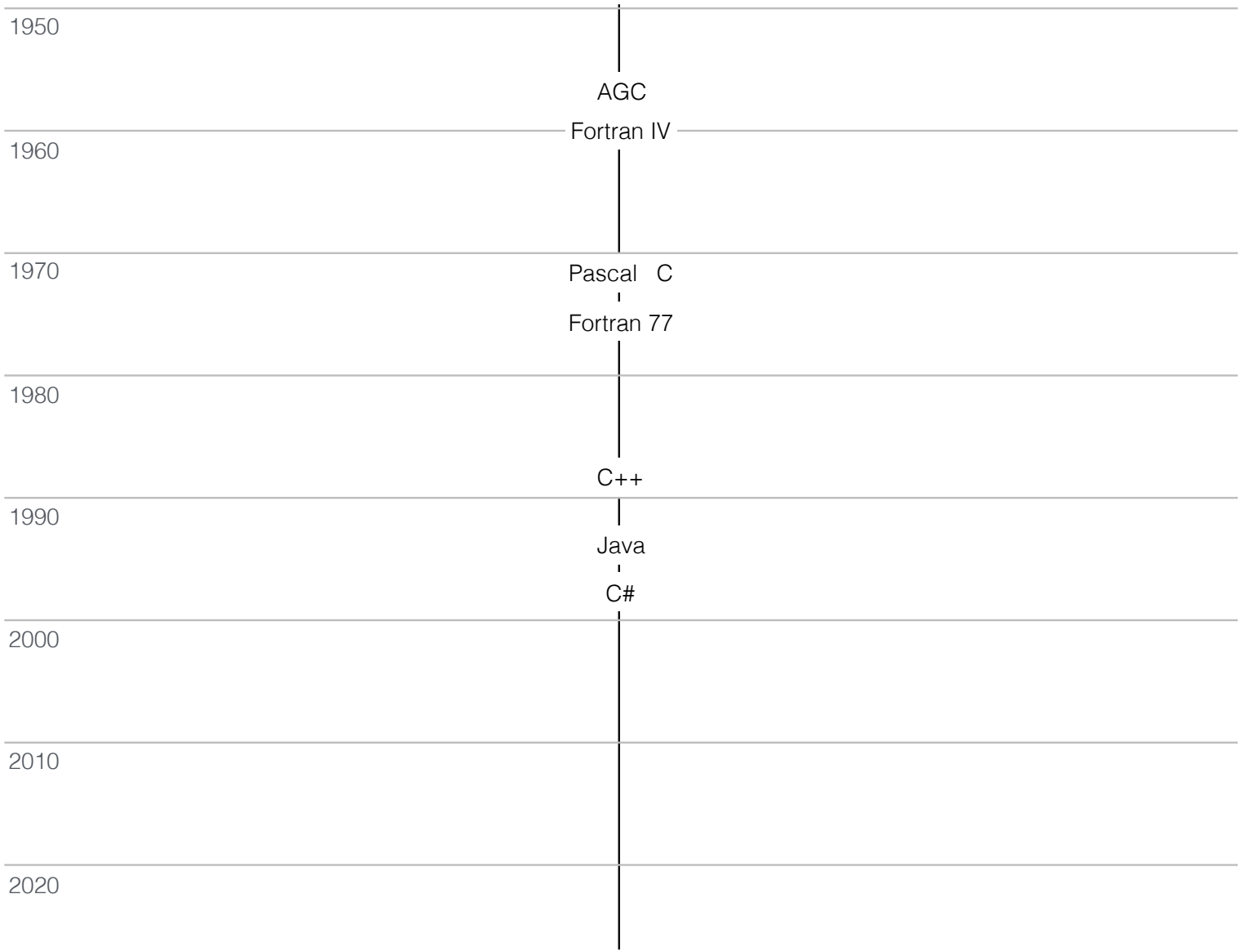
```
proc abs max = ([,]real a, ref real y, ref int i, k)real:

comment The absolute greatest element of the matrix a, of size ⌈a by 2⌈a
is transferred to y, and the subscripts of this element to i and k; comment

begin
   real y := 0; i := ⌊a; k := 2⌊a;
   for p from ⌊a to ⌈a do
     for q from 2⌊a to 2⌈a do
       if abs a[p, q] > y then
           y := abs a[p, q];
           i := p; k := q
       fi
     od
   od;
   y
end # abs max #
```

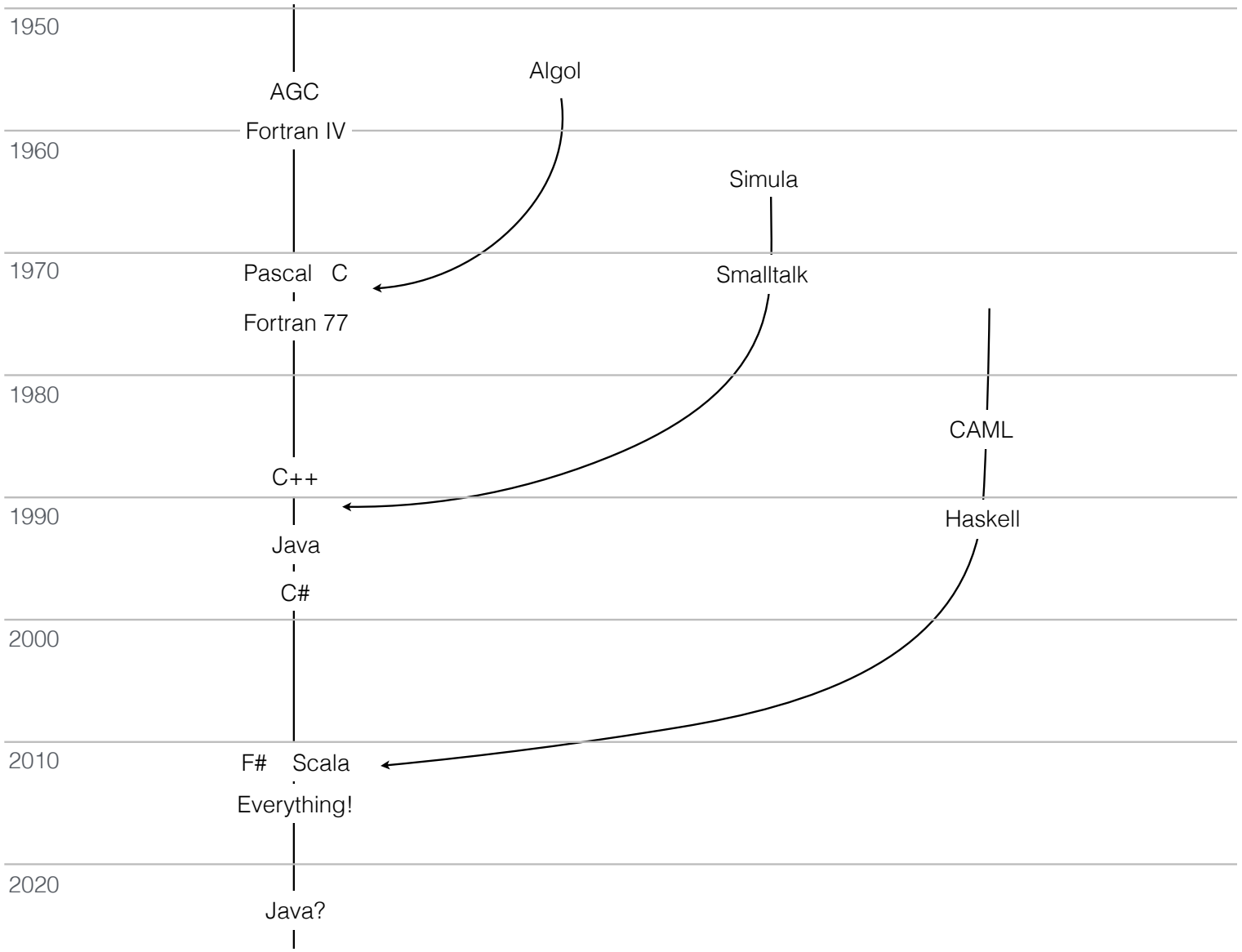
Simula

```
Class Rectangle (Width, Height); Real Width, Height;
                            ! Class with two parameters;
 Begin
    Real Area, Perimeter;  ! Attributes;

    Procedure Update;      ! Methods (Can be Virtual);
    Begin
      Area := Width * Height;
      Perimeter := 2*(Width + Height)
    End of Update;

    Boolean Procedure IsSquare;
      IsSquare := Width=Height;

    Update;                ! Life of rectangle started at creation;
    OutText("Rectangle created: "); OutFix(Width,2,6);
    OutFix(Height,2,6); OutImage
 End of Rectangle;
```

Smalltalk

```
SequenceableCollection extend [
  swap: i with: j [
    |t|
    t := self at: i.
    self at: i put: (self at: j).
    self at: j put: t.
  ]
].

Object subclass: Shuffler [
  Shuffler class >> Knuth: aSequenceableCollection [
    |n k|
    n := aSequenceableCollection size.
    [ n > 1 ] whileTrue: [
      k := Random between: 1 and: n.
      aSequenceableCollection swap: n with: k.
      n := n - 1
    ]
  ]
].
```


1950
AGC
Fortran IV
1960
1970
Pascal
C
Fortran 77
1980
C++
1990
Java
C#
2000
2010
2020


1950
1960
1970
1980
1990
2000
2010
2020
AGC
Fortran IV
Algol
Simula
Pascal
C
Smalltalk
Fortran 77
CAML
C++
Haskell
Java
C#
F#
Scala
Everything!
Java?

Closures

- What are they?
- What do they look like?
- What can you do with them?

What is a closure?

```
TrackInfo [] hitsOfThe70s = {…}

TrackInfo [] rankedHitsOfThe70s = hitsOfThe70s.sort ();
```

What is a closure?

```
knowItAllServer = Neilsen;
TrackInfo [] hitsOfThe70s = {…}

TrackInfo [] rankedHitsOfThe70s = hitsOfThe70s.sort (
   (a, b) => {
      return compare (
         knowItAllServer(a).totalSales,
         knowItAllServer(b).totalSales
      );
   }
);
```

How it looks

Pascal

```
PROGRAM Sort(input, output);
    CONST
        MaxElts = 50;
    TYPE
        IntArrType = ARRAY [1..MaxElts] OF Integer;
    VAR
    i, j, tmp, size: integer;
    arr: IntArrType;

        PROCEDURE ReadArr(VAR size: Integer; VAR a: IntArrType);
        BEGIN
            size := 1;
            WHILE NOT eof DO BEGIN
                readln(a[size]);
                IF NOT eof THEN
                    size := size + 1
            END
        END;

BEGIN
    ReadArr(size, arr);

    (* Sort using bubble sort. *)
    FOR i := size - 1 DOWNTO 1 DO
        FOR j := 1 TO i DO
            IF arr[j] > arr[j + 1] THEN BEGIN
                tmp := arr[j];
                arr[j] := arr[j + 1];
                arr[j + 1] := tmp;
            END;

    FOR i := 1 TO size DO
        writeln(arr[i])
END.
```

Pascal

```
Program A ();
    type fn : function (integer) : integer;

    var  a: int;
         f: fn;

    function B() : fn;

        function C(x: integer) : integer;

        begin
           a := a+x;
           return a;
        end;

    begin
        return @C;
    end;

begin
    a := 10;
    f := B();
    println f(5); { prints 15 }
end.
```

Javascript

```
function callAjax (url, callback) {
    var xmlhttp = new XMLHttpRequest();

    xmlhttp.onreadystatechange = function() {
        if (xmlhttp.readyState == XMLHttpRequest.DONE ) {
           if (xmlhttp.status == 200) {
               callback (xmlhttp.responseText);
           }
           else {
               alert('Error: ' + xmlhttp.status);
           }
        }
    };

    xmlhttp.open("GET", url, true);
    xmlhttp.send();
}

callAjax ("ajaxfile.txt", function (s) {
        document.getElementById("myDiv").innerHTML = s;
    }
)
```

Javascript

```
var serial = (function () {
    var counter = 0;
    var f = function () {
        return counter += 1;
    };
    return f;
}) ();

var x = serial ();
var y = serial ();
var z = serial ();
```

Python 3

```
def initCounter():
    x = 0

    def counter ():
        nonlocal x
        x += 1
        print x

    return counter

count = initCounter

count () # prints 1
count () # prints 2
count () # prints 3
```

C++11

```
#include stdio
#include functional
using std;

function <string (string)> times (int n) {
    auto ss = [&] (const string s) -> const string {
        stringstream os;
        for (int i=0; i<n; ++i) os << s;
        return os.str();
    }
    return ss;
}

int main () {
    auto f = times (5);
    cout << f (“Hello”);
    return 0;
}
```

C#

```
public static Func<int,int> GetAFunc() {
    var myVar = 1;

    Func<int, int> inc = delegate(int var1) {
         myVar = myVar + 1;
         return var1 + myVar;
    };
    return inc;
}


static void Main(string[] args) {
    var inc = GetAFunc();
    Console.WriteLine(inc(5));
    Console.WriteLine(inc(6));
}
```

~~Java 8~~

Scala

```
object Demo {
   def main(args: Array[String]) {
      println( "times(1) = " +  multiplier(1) )
      println( "times(2) = " +  multiplier(2) )
   }

   var factor = 3
   val times = (i:Int) => i * factor
}
```

What are they good for?

Iterators

```
Array <T> tt;
for (int k=0; k<tt.length(); ++k) // do something with tt[k]
```

```
Iterable <T> tt;
for (tt.reset(); !tt.atEnd(); tt.next()) // do something with tt.current()
```

```
Iterator <Iterable <T>> ti;
for (ti.reset (tt); !ti.atEnd(); ti.next()) // do something with ti.current()
```

```
foreach (t in tt) // do something with t
```

Iterators

```
Iterable <T> tt;
tt.foreach ( (t) -> { checkID (t) } );
```

```
gotcha = tt.findFirst ( (t) -> { return t.isTheDroidYoureLookingFor () } );
```

```
rebels = tt.filter ( (t) -> { return !t.isLoyalToDarth () } );
```

```
tt.filter ( (t) -> isLoyalToDarth() ).forEach ( (t) -> { arrest (t) } );
```

Iterators

```
class myTimetable:Map <Pair <Cday, Cslot>, List <Cevent> > {
      Cday days = [“Mon”, “Tue”, “Wed”, “Thu”, “Fri”];
      Cslot slots = [“9am”, “10am”, “11am”, “12pm”, “1pm”, “2pm”, “3pm”, “4pm”];

      string toHtml () {
            tm = new htmlTableMaker<CSlot, CDay> (slots, days);
            tm.foreach ( (s, d) → { return this [d, s]; } );
      }
}

new myTimetable( … ).toHtml ();
```

```
class htmlTableMaker <X,Y> {
     ctor (X x, Y y)  { xx=x; yy=y };

     string foreach ((X x, Y y) → Iterable <Z> f) =
            “<table>
                  <tr> {
                        ([“”] + yy).foreach (y → “<th> {y} </th>”) // header row
                  } </tr> {
                        xx.foreach ( (x) → {
                              “<tr>
                                    <th> {x} </th> { // header column
                                          yy.foreach ( (y) → {
                                                “<td> {
                                                      f(x,y).foreach ( (q) → {
                                                            “<p> { q } </p>”
                                                      } );
                                                } </td>”;
                                          );
                                    }
                              </tr>”;
                        );
                  }
            </table>”;
}
```

Autonomy

```
class Caller inherits Dialog {
    Telephony tp;

    bool hangupFlag;
    Textbox number;
    Text status;

    Button call.onClick ( () -> {
        hangupFlag = false;
        mutex q = telephphony.phoneline; q.acquire ();
        tp = new telephony().dial (number value);
        while (!hangupFlag) sleep (1000);
        tp.hangup ();
        q.release ();
    } );

    Button hangup.onClick ( () -> { hangupFlag = true } );
}
```

Autonomy

```
class Caller inherits Dialog {
    Telephony tp;
    Thread t;

    Textbox number;
    Text status;

    private void callerTask (string number) {
        static mutex q = telephony.phoneLine; q.acquire ();
        tp = new telephony();
        tp.dial (number);
        while (tp.isCallInProgress) sleep (1000);
        q.release ();
    }

    Button call.onClick ( () -> {
        t = new Thread (callerTask).start (number.value);
    } );

    Button hangup.onClick ( () -> { tp.hangup () } );
}
```

Javascript

```
function callAjax (url, callback) {
    var xmlhttp = new XMLHttpRequest();

    xmlhttp.onreadystatechange = function() {
        if (xmlhttp.readyState == XMLHttpRequest.DONE ) {
           if (xmlhttp.status == 200) {
               callback (xmlhttp.responseText);
           }
           else {
               alert('Error: ' + xmlhttp.status);
           }
        }
    };

    xmlhttp.open("GET", url, true);
    xmlhttp.send();
}

callAjax ("ajaxfile.txt", function (s) {
        document.getElementById("myDiv").innerHTML = s;
    }
)
```

Autonomy

```
class Caller inherits Dialog {
    Telephony tp;
    Thread t;

    Textbox number;
    Text status;

    private void callerTask (string number) {
        static mutex q = telephony.phoneLine; q.acquire ();
        tp = new telephony();
        tp.dial (number);
        while (tp.isCallInProgress) sleep (1000);
        q.release ();
    }

    Button call.onClick ( () -> {
        t = new Thread (callerTask).start (number.value);
    } );

    Button hangup.onClick ( () -> { tp.hangup () } );
}
```

Autonomy

```
class Caller inherits Dialog {
    TextBox number;
    Text status;

    button call.onClick ( () => { tp.placeCall (number.value); } );

    button hangup.onClick ( () -> { tp.hangup () } );enable (false);

    TelephonySane tp ()
      .onStartDial ( (string s) => {
            status.value = "Calling: " + s;
        }   call.enable = false; hangup.enable = true;
      .onConnect ( () -> {
      .onConnect ( () => {
            status.value = "Connected";
        }   status.value = "Connected";
      .onDisconnect ( () -> {
      .onDisconnect ( () => {
            status.value = "Ended";
        }   status.value = "Ended";
      .onError ( (string msg) => {
            call.enable = false; hangup.enable = true;
        }   status.value (msg).style(red);
      .onError ( (string msg) -> {
      .priority (normal);
            status.value (msg).style(red);
}       }
      .priority (normal);
}
```

Questions?