Universitas Esa Unggul

Smart, Creative and Entrepreneurial

www.esaunggul.ac.id

CMC 101 TOPIK DALAM PEMROGRAMAN

PERTEMUAN 1

PROGRAM STUDI MAGISTER ILMU KOMPUTER

FAKULTAS ILMU KOMPUTER

TOPIK DALAM PEMROGRAMAN PARADIGMA PEMROGRAMAN

Pertemuan 2

TUJUAN PERKULIAHAN

- Mahasiswa mengenal beberapa paradigma pemrograman dan perbedaan antar paradigma
- Definisi dari paradigma pemrograman
- Pengenalan paradigma pemrograman prosedural, fungsional, deklaratif, berorientasi objek, *event-driven/ reactive*

Programming language concept

Programming Languages

- Low Level vs High Level
- Machine language, Assembly
- Compile vs Interpret
- Procedural vs Functional vs Object Oriented
- Programming Languages, past and present

Low Level vs High Level

- Low Level languages
- Easy for the computer machine to understand
- Very difficult for humans to work with.

Low Level vs High Level

- High Level
- Easy for humans to read
- Impossible for machines
- Must be translated into machine language using an interpreter or compiler

Machine Language

- Instructions executed directly by a computer's central processing unit (CPU)
- Every program directly executed by a CPU is made up of a series of machine language instructions.

Assembly Language

- Converted into machine code by an *assembler*
- Conversion process is referred to as *assembly*

```
section .text

    global _start

_start:

        ; write our string to stdout.

        mov     edx,len    ; third argument: message length.
        mov     ecx,msg    ; second argument: pointer to message to write.
        mov     ebx,1      ; first argument: file handle (stdout).
        mov     eax,4      ; system call number (sys_write).
        int     0x80       ; call kernel.

        ; and exit.

        mov     ebx,0      ; first syscall argument: exit code.
        mov     eax,1      ; system call number (sys_exit).
        int     0x80       ; call kernel.

section .data

msg     db      "Hello, world!",0xa      ; the string to print.
len     equ     $ - msg                  ; length of the string.
```

Source file for "Hello World" would look like this:

```
b8    21 0a 00 00   #moving "!\n" into eax
a3    0c 10 00 06   #moving eax into first memory location
b8    6f 72 6c 64   #moving "orld" into eax
a3    08 10 00 06   #moving eax into next memory location
b8    6f 2c 20 57   #moving "o, W" into eax
a3    04 10 00 06   #moving eax into next memory location
b8    48 65 6c 6c   #moving "Hell" into eax
a3    00 10 00 06   #moving eax into next memory location
b9    00 10 00 06   #moving pointer to start of memory location into ecx
ba    10 00 00 00   #moving string size into edx
bb    01 00 00 00   #moving "stdout" number to ebx
b8    04 00 00 00   #moving "print out" syscall number to eax
cd    80            #calling the linux kernel to execute our print to stdout
b8    01 00 00 00   #moving "sys_exit" call number to eax
cd    80            #executing it via linux sys_call
```

Low Level vs High Level

Low level

Machine language

Assembly language

High Level

Fortran	Pascal
Cobol	C++
BASIC	Java

Compile vs Interpret

Interpret

- Translate the program line by line, running the program a step at a time
- Like a conversation between two people who speak in different languages

Compile vs Interpret

Compile

- Translate the entire program first, then run it
- Like translating an entire letter or document from one language to another, then reading it in the new language.

Programming Language Paradigms

- Procedural
- Functional
- Object Oriented

Procedural Programming

- Very structured
- Step by Step Procedures
- Uses loops to iterate
- Fortran
- Cobol
- C

Functional Programming

- Strongly math based
- Uses functions to calculate answers based on given data
- Uses recursion to iterate
- LISP
- Haskell
- Scheme

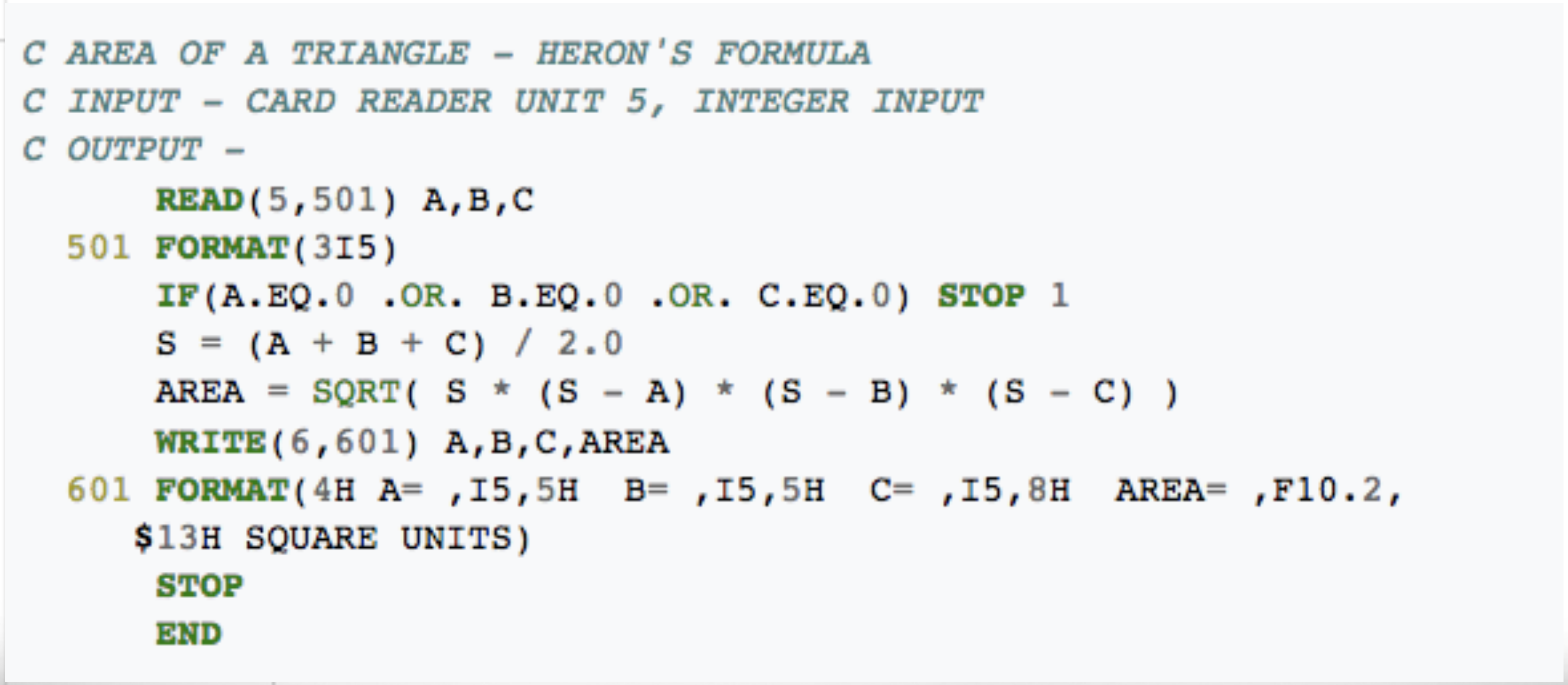
Object Oriented Programming

- Primary focus is on the data
- Actions and processes strongly tied to the data
- Uses both loops and recursion
- Uses objects, classes, abstraction, inheritance, and polymorphism
- C++, Java, Python

Fortran

- FORmula TRANslation – the first significant high level language
- Developed by John Backus in the mid 50s
- *"Much of my work has come from being lazy. …so,…I started work on a programming system to make it easier to write programs."*

- comments must begin with a * or C or ! in column 1
- statement labels must occur in columns 1-5
- continuation lines must have a non-blank character in column 6
- statements must start in column 7
- the line-length may be limited to 72 characters (derived from the 80-byte width of a punch-card, with last 8 characters reserved for (optional) sequence numbers)



```
C AREA OF A TRIANGLE - HERON'S FORMULA
C INPUT - CARD READER UNIT 5, INTEGER INPUT
C OUTPUT -
      READ(5,501) A,B,C
  501 FORMAT(3I5)
      IF(A.EQ.0 .OR. B.EQ.0 .OR. C.EQ.0) STOP 1
      S = (A + B + C) / 2.0
      AREA = SQRT( S * (S - A) * (S - B) * (S - C) )
      WRITE(6,601) A,B,C,AREA
  601 FORMAT(4H A= ,I5,5H  B= ,I5,5H  C= ,I5,8H  AREA= ,F10.2,
     $13H SQUARE UNITS)
      STOP
      END
```

```
C     FORTRAN IV WAS ONE OF THE FIRST PROGRAMMING
C     LANGUAGES TO SUPPORT SOURCE COMMENTS
      WRITE (6,7)
    7 FORMAT(13H HELLO, WORLD)
      STOP
      END
```

COBOL

- **CO**mmon **B**usiness **O**riented **L**anguage
- Developed in the late 50s largely based on designs by Grace Hopper
- Designed to work better with programs developed for business applications

```
$ vim helloworld


IDENTIFICATION DIVISION.
PROGRAM-ID. HELLO-WORLD.
* simple hello world program
PROCEDURE DIVISION.
    DISPLAY 'Hello world!'.
    STOP RUN.
```

```
      $ SET SOURCEFORMAT"FREE"
IDENTIFICATION DIVISION.
PROGRAM-ID.  InputSort.
AUTHOR.  Michael Coughlan.
* An example program using the SORT and an
* INPUT PROCEDURE.  The program accepts records
* from the user and RELEASEs them to the work file
* where they are sorted.  This program
* allows student records to be entered in any order but
* produces a file sequenced on ascending StudentId.

ENVIRONMENT DIVISION.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT StudentFile ASSIGN TO "SORTSTUD.DAT"
                ORGANIZATION IS LINE SEQUENTIAL.
    SELECT WorkFile ASSIGN TO "WORK.TMP".


DATA DIVISION.
FILE SECTION.
FD StudentFile.
01 StudentDetails      PIC X(30).

* The StudentDetails record has the description shown below.
* But in this program we don't need to refer to any of the items in
* the record and so we have described it as PIC X(32)
* 01 StudentDetails
*     02  StudentId        PIC 9(7).
*     02  StudentName.
*         03 Surname       PIC X(8).
*         03 Initials      PIC XX.
*     02  DateOfBirth.
*         03 YOBirth       PIC 9(4).
*         03 MOBirth       PIC 9(2).
*         03 DOBirth       PIC 9(2).
*     02  CourseCode       PIC X(4).
*     02  Gender           PIC X.
```

```
SD WorkFile.
01 WorkRec.
   02 WStudentId        PIC 9(7).
   02 FILLER            PIC X(23).


PROCEDURE DIVISION.
Begin.
   SORT WorkFile ON ASCENDING KEY WStudentId
        INPUT PROCEDURE IS GetStudentDetails
        GIVING StudentFile.
   STOP RUN.


GetStudentDetails.
    DISPLAY "Enter student details using template below."
    DISPLAY "Enter no data to end.".
    DISPLAY "Enter - StudId, Surname, Initials, YOB, MOB, DOB, Course, Gender"
    DISPLAY "NNNNNNNSSSSSSSSIIYYYYMMDDCCCCG"
    ACCEPT  WorkRec.
    PERFORM UNTIL WorkRec = SPACES
       RELEASE WorkRec
       ACCEPT WorkRec
    END-PERFORM.
```

BASIC

- Beginner's All-purpose Symbolic Instruction Code
- Dartmouth College, New Hampshire
- Family of general-purpose, high-level programming languages
- Design philosophy emphasizes ease of use.

```
10 PRINT "Hello World!"
20 GOTO 10
```

Pascal

- Procedural programming language
- Named in honor of the 17th century French mathematician Blaise Pascal
- Designed in the late 60s by Niklaus Wirth.
- Good for teaching proper programming techniques

```
program HelloWorld;

begin
  writeln('Hello World');
end.
```

C++

- C++ is a general-purpose programming language
- Designed in the late 70s by Bjarne Stroustrup
- Extension to the **C** language with object-oriented data abstraction mechanisms.

```
#include <iostream.h>

main()
{
     cout << "Hello World!";
     return 0;
}
```

Java

- General-purpose computer programming language
- Class-based and object-oriented
- Able to run on any platform because of the JVM (Java Virtual Machine)

Java

- Java source code programs (the .java file) are compiled into **bytecode**, (the .class file)
- Bytecode is a universal “middle level” language code
- Very close to machine level
- Translated by the JVM

```
public class HelloWorld {
   public static void main(String[] args) {
      // Prints "Hello, World" in the terminal window.
      System.out.println("Hello, World");
   }
}
```

Python

- Widely used general-purpose, high-level programming language
- Design philosophy emphasizes code readability
- Syntax allows programmers to express concepts in fewer lines of code

hello.py

```
print("Hello, World!")
```

Summary

- Low Level vs High Level
- Compile vs Interpret
- Procedural vs Functional vs Object Oriented
- Programming Languages, past and present

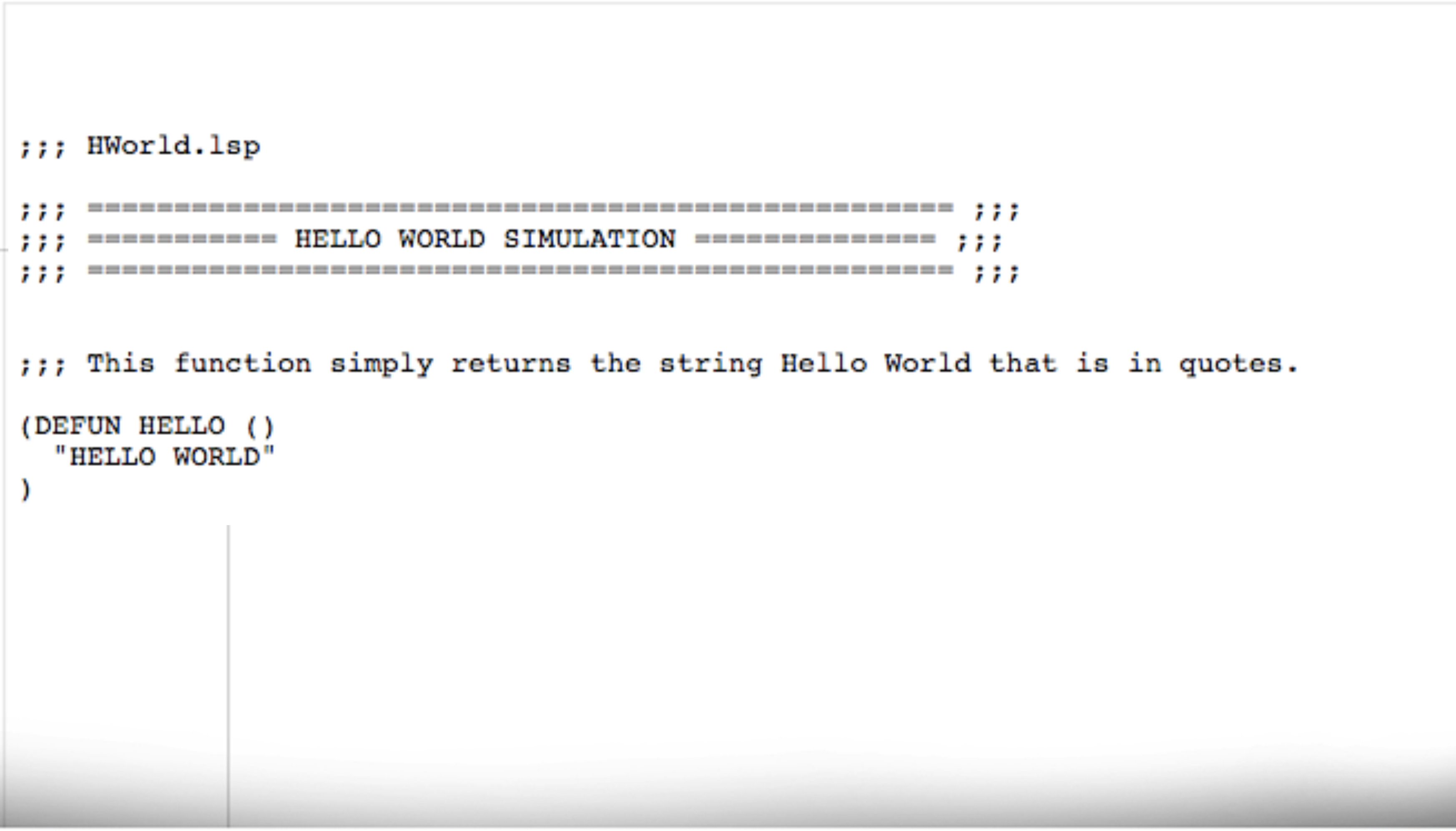
Perl

```
#!/usr/bin/perl
#
# The traditional first program.

# Strict and warnings are recommended.
use strict;
use warnings;

# Print a message.
print "Hello, World!\n";
```

Lisp



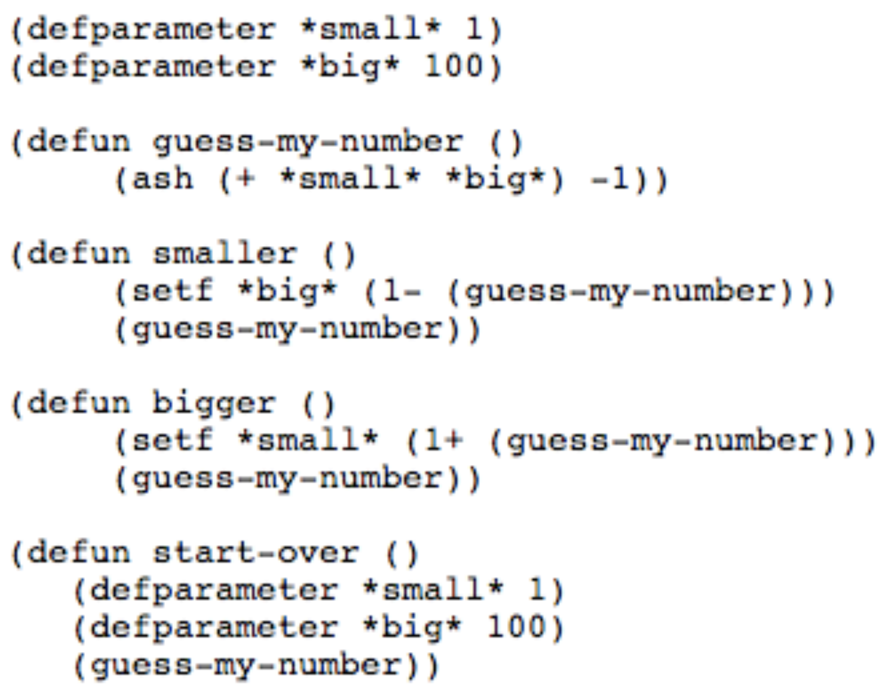
```
;;; HWorld.lsp

;;; ================================================== ;;;
;;; =========== HELLO WORLD SIMULATION ============== ;;;
;;; ================================================== ;;;


;;; This function simply returns the string Hello World that is in quotes.

(DEFUN HELLO ()
  "HELLO WORLD"
)
```

Lisp (guessing number)



```
(defparameter *small* 1)
(defparameter *big* 100)

(defun guess-my-number ()
     (ash (+ *small* *big*) -1))

(defun smaller ()
     (setf *big* (1- (guess-my-number)))
     (guess-my-number))

(defun bigger ()
     (setf *small* (1+ (guess-my-number)))
     (guess-my-number))

(defun start-over ()
   (defparameter *small* 1)
   (defparameter *big* 100)
   (guess-my-number))
```

TERIMA KASIH