LEC1 OOP 2018-2019

Programming Paradigms

Programming paradigm is a way to classify programming languages according to their style of programming and features they provide. There are several features that determine a programming paradigm such as modularity, objects, interrupts or events, control flow etc. A programming language can be single paradigm or multi-paradigm.

With the wide spread of programming, various programming paradigms came into existence. Some popular programming paradigms are:

1- Imperative programming

Imperative programming is the oldest paradigm and is still in practice. It is the widely practiced paradigm in the day-to-day programming. It mainly focuses on steps to be done and works on the logic of "*First do this then do that*". It defines a sequence of statements in order of which the operations must take place. In imperative programming, the control flow is explicit and depend on collection of **GOTO** statements. Imperative programming lacks the support of modularity.

Examples of imperative programming languages are: Assembly, C, C++, Java etc.

Below is the imperative programming approach of listing the first and last name of students from a list **student** whose marks is greater than 90.

```
names = []
i = 0
totalStudents = length(student)
start:
   if i >= totalStudents then goto end
   current = student[i]
   marks = current.marks
   if marks < 90 goto next
   addToList(names, current.firstname + current.lastname)
next:
   i = i + 1
   goto start
end:
   print names
```

2- Declarative Programming

Declarative programming focuses on the **logic of computation neglecting the control flow**. It specifies what the result should be, without specifying how to obtain the result. Declarative programming generally does not contain if-else, loops and other control flow statements. Imperative programming defines each minute steps to get the result, whereas in contrast declarative programming only defines the logic of computation.

Popular declarative programming languages are: SQL, XQuery

Below is the declarative programming approach to get first and last name of students from a list **student** whose marks is greater than 90.

```
select firstname, lastname
from student
where student.marks >= 90
```

3- Structured Programming

Structured programming is a kind of imperative programming, focuses on modular programming ***{ Definition Modular Programming : Modular programming is the process of subdividing a computer program into separate sub-programs.***

A module is a separate software component. It can often be used in a variety of applications and functions with other components of the system. Similar functions are grouped in the same unit of programming code and separate functions are developed as separate units of code so that the code can be reused by other applications }.

It makes extensive use of for, while, subroutines, blocks and sub-blocks, rather than simply using **GOTO** which leads to complex and tangled code. Structured programming was introduced as an improvement over the imperative programming approach to get more control over the program.

Examples of structured programming languages are – C, C++, Pascal, Ada etc.

Below is the structured programming approach to get first and last name of students from a list **student** whose marks is greater than 90.

```
names = []

for i = 0, i <= length(student), i = i + 1 {
    if student[i].marks >= 90 {
        addToList(names, student[i].firstname, student[i].lastname)
    }
}

for i = 0, i <= length(names), i = i + 1 {
    print(names[i])
}
```

4- Procedural Programming

Procedural programming is derived from structured programming. It inherits all properties of structured and imperative programming. Procedural programming is based on procedural calls. Each statement of a procedural language is either procedure call or data assignment.

Note: *Procedures, routines, subroutines and functions all are same thing with little difference.*

Some popular procedural programming languages are: C, Pascal, BASIC, Fortran

Below is the procedural programming approach to get first and last name of students from a list **student** whose marks is greater than 90.

```
void main() {
   names = getStudentNames()
   printList(names) }
names getStudentNames() {
   names = []
   for (i = 0; i <= length(student); i = i + 1) {
      if (student[i].marks >= 90 ) {
         addToList(names, student[i].firstname, student[i].lastname)
      }    }
```

```
    return names}
void printList(names) {
    for (i = 0; i <= length(names); i = i + 1) {
        print(names[i])    } }
```

5- Object Oriented Programming (OOP)

Object Oriented Programming paradigm is widely practiced programming paradigm. It is based on the concept of objects. Objects are real world entity. Everything present around us is an object. Every object has two important property attribute (data) and behavior (function).

For example:

Car is an object having attributes : type, color, model etc.

Behavior: turnLeft(), turnRight(), back() etc.

Object-Oriented programming is also inherited from imperative and procedural programming.

Popular object oriented programming languages are – Simula-67, Java, C++, C# etc.

Below is the object oriented programming approach to get first and last name of students from a list **student** whose marks is greater than 90.

```
for s in student {
      if s.marks >= 90 {
            print(s.firstname + s.lastname);
      }
}
```

6- Logic Programming

Logic programming is a type of programming paradigm which is largely based on formal logic. Any program written in a logic programming language is a set of sentences in logical form, expressing facts and rules about some problem domain.

7- Functional Programming

Functional programming paradigm is completely different programming approach from all paradigms mentioned above. Functional programming uses a combination of functions calls to drive the flow of the program. The result of a function becomes the input to another function.

Popular functional programming languages are – Python, Lisp, Clojure, Haskell etc.