PROGRAMMING PARADIGMS:

There are basically four programming paradigms in programming languages:

1) Imperative programming languages
2) Functional programming languages
3) Logic programming languages
4) Object-oriented programming languages

Others are:

5) Event-Driven programming languages
6) Concurrent / parallel programming languages
7) Special purpose programming languages

1) Imperative programming languages:
It is also known as procedural languages.
An imperative language uses a sequence of statements to determine how to reach a certain goal. Means here you will get the answer how to do a task not what to do.
For example in C,

```
int a = 4;
int b = 5;
int sum = 0;
sum = a + b;
```

From assigning values to each variable to the final addition of those values,each statement changes the state of the program. Using a sequence of five statements the program shows how to add the numbers 4 and 15.

Example of Imperative language is: C, C++

2) Functional programming languages:
It is also known as applicative languages.
Functional programming is a form of declarative programming (mathematical function).
Programming consists of building the function that computes the answer.
There are two types:

1. Pure Functional languages: It supports only functional paradigm. Ex. Haskell.
2. Impure Functional languages: It supports both functional and imperative paradigm. Ex. Lisp.

Example of Functional language is: Lisp, Python, Haskell.

3) Logic programming languages:
This type of languages concentrate on what is the expected outcome for the program instead of how the outcome is achieved. Logical programming is something like math .Logic program statements express facts and rules about problems. To understand the rules, lets take an example like, “A is true if B and C is true”.
And to understand facts we can say that “A is true”.

Example of Logic language is: Prolog.

4) Object-oriented programming languages:
Object oriented programming language based on object instead of just functions and procedures. It involves concepts of oops programming languages.
For example: C++, Java.

5) Event Driven programming languages:
These languages are execute various operations based on user activities like mouse click and other events.
For example: Visual Basic, Java.
6) Concurrent or Parallel programming languages:
These are used to build various distributed programs.
For example: SR, Linda.
7) Special purpose programming languages:
These programming languages are used for special task.
For example: SQL, HTML.

Click here to view on YouTube
MCQs:

Q1. Which of the following is not a programming paradigm style?
a) Functional paradigm
b) Imperative paradigm
c) Procedure paradigm
Q2. Logic paradigm is largely utilized in_____.
a)Artificial Intelligence
b)IOT
c)Web Development
Q3. In object-oriented, everything is represented as an_____.
a)Functions
b)Objects
c)constructor
Q4. Popular functional programming languages are_____.

a) Python

b) Lisp

c) C++

Q5. The ‘first do this, next do that’ is a short phrase for describing which style of programming paradigm?

a) Imperative

b) Functional

c) Logic

Q6. We can simply define programming paradigm as_____

a) A programming language

b) A process

c) A classifying style of a programming language

Q7. While the functional paradigm emphasizes the idea of a mathematical function, the logic paradigm focusses on which of the following?

a) Objects

b) Classes

c) Predicate logic

Q8. Which of the following is not a characteristic of object oriented programming paradigm?

a) Abstraction

b) Inheritance

c) Subprogram

Q9. The Functional programming paradigm is mainly used to perform mathematical functions without changing_____

a) State

b) Function

c) Program

Q10. What is the second name of imperative programming Paradigm?

a) Procedural Paradigm

b) Functional Paradigm

c) Object Oriented Paradigm

MCQs Answers:

Q1. (c)

Q2. (a)

Q3. (b)

Q4. (c)

Q5. (a)

Q6. (c)

Q7. (c)

Q8. (c)

Q9. (a)

Q10. (a)

Principles of Programming Languages:

EasyExamNotes.com covered following topics in these notes.

- Language Evaluation Criteria
- Influences on Language Design
- Language Categories
- Programming Paradigms
- Compilation
- Virtual Machines
- Programming Environments
- Issues in Language Translation
- Parse Tree

- Pointer and Reference type
- Concept of Binding
- Type Checking
- Strong typing
- Sequence control with Expression
- Exception Handling
- Subprograms
- Fundamentals of sub-programs
- Scope and lifetime of variable
- static and dynamic scope
- Design issues of subprogram and operations
- Local referencing environments
- Parameter passing methods
- Overloaded sub-programs
- Generic sub-programs
- Design issues for functions
- co routines
- Abstract Data types
- Abstraction and encapsulation
- Static and Stack-Based Storage management
- Garbage Collection
- OOP in C++
- OOP in Java
- OOP in C#
- OOP in PHP
- Concurrency
- Semaphores
- Monitors

- Message passing
- Java threads
- C# threads
- Exception handling
- Exceptions
- Exception Propagation
- Exception handler in C++
- Exception handler in Java
- Introduction and overview of Logic programming
- Basic elements of Prolog
- Application of Logic programming
- Functional programming languages
- Introduction to 4GL

Practicals:

- Memory Implementation of 2D Array.
- Memory Implementation of 3D Array.
- Implementation of pointers in C++.
- Write a program in Java to implement exception handling.
- Write a program in C++ to implement call by value parameter passing Method.
- Write a program in C++ to implement call by reference parameter passing Method.
- Write a program in Java to implement concurrent execution of a job using threads.
- Implement Inheritance in C#.
- Implement Encapsulation in C#.
- Implement static/compiletime Polymorphism in C#.

- Implement dynamic/runtime Polymorphism in C#.

Previous years solved papers:

- PPL|RGPV|May 2018
- PPL|RGPV|June 2017

A list of Video lectures

- Click here

References:

1. Sebesta,"Concept of programming Language", Pearson Edu
2. Louden, "Programming Languages: Principles & Practices" , Cengage Learning
3. Tucker, "Programming Languages: Principles and paradigms ", Tata McGraw –Hill.
4. E Horowitz, "Programming Languages", 2nd Edition, Addison Wesley

Related posts:

1. Sequence Control & Expression | PPL
2. PPL:Named Constants
3. Parse Tree | PPL | Prof. Jayesh Umre
4. Basic elements of Prolog
5. Loops | PPL | Prof. Jayesh Umre

6. Subprograms Parameter passing methods | PPL | Prof. Jayesh Umre
7. Programming Paradigms | PPL | Prof. Jayesh Umre
8. Subprograms Introduction | PPL | Prof. Jayesh Umre
9. Phases of Compiler | PPL | Prof. Jayesh Umre
10. Parse Tree | PPL
11. Influences on Language design | PPL | Prof. Jayesh Umre
12. Fundamentals of Subprograms | PPL | Prof. Jayesh Umre
13. Programming Paradigm
14. Influences on Language Design
15. Language Evaluation Criteria
16. OOP in C++ | PPL
17. OOP in C# | PPL
18. OOP in Java | PPL
19. PPL: Abstraction & Encapsulation
20. PPL: Semaphores
21. PPL: Introduction to 4GL
22. PPL: Variable Initialization
23. PPL: Conditional Statements
24. PPL: Array
25. PPL: Strong Typing
26. PPL: Coroutines
27. PPL: Exception Handler in C++
28. PPL: OOP in PHP
29. PPL: Character Data Type
30. PPL: Exceptions
31. PPL: Heap based storage management
32. PPL: Primitive Data Type

33. PPL: Data types
34. Programming Environments | PPL
35. Virtual Machine | PPL
36. PPL: Local referencing environments
37. Generic Subprograms
38. Local referencing environments | PPL | Prof. Jayesh Umre
39. Generic Subprograms | PPL | Prof. Jayesh Umre
40. PPL: Java Threads
41. PPL: Loops
42. PPL: Exception Handling
43. PPL: C# Threads
44. Pointer & Reference Type | PPL
45. Scope and lifetime of variable
46. Design issues for functions
47. Parameter passing methods
48. Fundamentals of sub-programs
49. Subprograms
50. Design issues of subprogram
51. Garbage Collection
52. Issues in Language Translation
53. PPL Previous years solved papers
54. Type Checking | PPL | Prof. Jayesh Umre
55. PPL RGPV May 2018 solved paper discussion| Prof. Jayesh Umre
56. PPL Viva Voce
57. PPL RGPV June 2017 Solved paper | Prof. Jayesh Umre
58. Concurrency
59. Basic elements of Prolog

60. Introduction and overview of Logic programming
61. Application of Logic programming
62. PPL: Influences on Language Design
63. Language Evaluation Criteria PPL
64. PPL: Sequence Control & Expression
65. PPL: Programming Environments
66. PPL: Virtual Machine
67. PPL: Pointer & Reference Type
68. try-catch block in C++