Influences on Language Design

- 1. Computer Architecture
- 2. Programming Methodologies
- 3. Virtual Machine

1. Computer Architecture:



Computer architecture in based on Von Neumann Architecture. A programming language is also affected by the architecture of computer. But how ? This is the question will see here. When we deploy/run a programs it reside in a memory and executed by the CPU. There are registers like program counter, instruction register etc. Each instruction going from memory to CPU is decided by program counter. And program counter get instruction info from instruction register. In this a way a program developed in languages passes trough this kind of cycle. Which affect the execution time of the language.

If system is multi user it will consume lot of time.

2. Programming Methodologies:

 Structured programming: This programming methodology also called top own design and step wise refinement. Its deficiency was incompleteness of type checking and inadequacy of control statements, which requires extensive use of go to statements. Ex. C.

- Data oriented programming: It focuses on data oriented methods emphasize data design, focusing on the use of abstract data types to solve the problem. Ex. Simula.
- Procedure oriented programming: It is the opposite of data oriented programming. Ex. C#.
- Object oriented programming: It gives support for Data abstraction, inheritance, polymorphism. Ex. C++.

3. Virtual Machine:

Virtual machines are software's on which other software's can be executed as they are executing on a physical machine.

There are two types of virtual machine:

- 1. Application/ process virtual machine.
- 2. System/hardware virtual machine.

1. Application/process virtual machine: In this we can take example of JVM. Because of JVM, Java is a platform independent language.

2. System/hardware virtual machine: In this we can take example of Virtual Box. Which gives as ability to run multiple of operating systems on a single physical machine. Here operating system is also a programming language which should be supported by Virtual Box.

Click here to view on YouTube

Viva Vice Q & A:

Q1. which factors influence the language?

Ans: virtual machine, comp. architecture, user, programming methodology, etc.

Q2. Program which interprets each line of high level program at time of execution is called?

Ans: Interpreter.

Q3. Program which interprets whole para of high level program at time of execution is called? Ans: Compiler.

Q4. During the execution of the instructions, a copy of the instructions is placed in which part?

Ans: Cache.

Q5. Functions that are used in programs and are defined by programmers are called.

Ans: User-defined functions.

Q6. The internal Components of the processor are connected by?

Ans: Bus.

Q7. Set of data whose items are organized together is classified as.....

Ans: data structure.

Q8. The _____ format is usually used to store data.

Ans: BCD.

Q9. To reduce the memory access time we generally make use of ?

Ans: Cache.

Q10. The computer's own binary-based language is also known as?

Ans: Machine language.

Q11. Write example of "third generation" programming languages (3GL) ?

Ans: FORTRAN, LISP, and COBOL.

Q12. What is the common property of logic programming languages and functional languages is?

Ans: Both are declarative.

Q13. Programming language 'FORTRAN' stands for

Ans: formula translator

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. Language Evaluation Criteria PPL
- 63. PPL: Sequence Control & Expression
- 64. PPL: Programming Environments
- 65. PPL: Virtual Machine
- 66. PPL: Programming Paradigm
- 67. PPL: Pointer & Reference Type
- 68. try-catch block in C++