

PRIMITIVE DATA TYPE

Primitive data types are data types which are not defined with support of other data types.

Some of the primitive data types are:

1. Numeric types
2. Boolean types
3. Character types

1. **Numeric Types:** These are the numbers.

1. **Integer:** A data type that represents some range of mathematical integers. Different programming languages supports different size of integers.

For example,

- Java includes four signed integer sizes: byte, short, int, and long.
- C++ and C#, include unsigned integer types (value without sign).
- `int age = 20;`

2. **Floating-Point:** Floating-point data types model real numbers. Most languages include two floating-point types, often called float and double.

- Float type is the standard size, usually being stored in four bytes of memory.
- Double type is provided for situations where larger fractional parts and/or a larger range of exponents is needed.

The collection of values that can be represented by a floating-point type is defined in terms of precision and range.

- Precision is the accuracy of the fractional part of a value, measured as the number of bits.
- Range is a combination of the range of fractions and, more important, the range of exponents.

3. **Complex:** Complex values are represented as ordered pairs of floating-point

values. It contains a imaginary number. It includes operations for arithmetic on complex values.

For example, in Python, $(8 + 2j)$, contains j as imaginary number.

4. **Decimal:** Decimal data types store a fixed number of decimal digits. For example, the number 0.1.
2. **Boolean Types:** It has only two elements, true and false. Boolean types are often used to represent switches or flags in programs.
3. **Character Types:** It stores a single character and requires a single byte of memory in almost all compilers.

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: Data types
33. Programming Environments | PPL
34. Virtual Machine | PPL
35. PPL: Local referencing environments
36. Generic Subprograms
37. Local referencing environments | PPL | Prof. Jayesh Umre
38. Generic Subprograms | PPL | Prof. Jayesh Umre
39. PPL: Java Threads
40. PPL: Loops
41. PPL: Exception Handling

42. PPL: C# Threads
43. Pointer & Reference Type | PPL
44. Scope and lifetime of variable
45. Design issues for functions
46. Parameter passing methods
47. Fundamentals of sub-programs
48. Subprograms
49. Design issues of subprogram
50. Garbage Collection
51. Issues in Language Translation
52. PPL Previous years solved papers
53. Type Checking | PPL | Prof. Jayesh Umre
54. PPL RGPV May 2018 solved paper discussion| Prof. Jayesh Umre
55. PPL Viva Voce
56. PPL RGPV June 2017 Solved paper | Prof. Jayesh Umre
57. Concurrency
58. Basic elements of Prolog
59. Introduction and overview of Logic programming
60. Application of Logic programming
61. PPL: Influences on Language Design
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++