ARRAY

Array: An array is a homogeneous collection of data elements in which an individual

element is identified by its position in the, relative to the first element.
The individual data elements of an array are of the same type.

- In the languages, such as C, C++, Java, Ada, and C#, all of the elements of an array are required to be of the same type.
- In some other languages, such as JavaScript, Python, and Ruby, variables are typeless references to objects or data values. In these cases, arrays still consist of elements of a single types, but the elements can reference objects or data values of different types. Such arrays are still homogeneous, because the array elements are of the same type.

type, but the elements can reference objects or data values of different types.

Some of the types of array:

1. Single Dimensional Array
2. Two Dimensional Array
3. Three Dimensional array

What are the design issues for arrays?

The design issues for arrays are:

- What type are legal for subscripts?
- When are subscript ranges bound?
- When does array allocation takes place?
- Are ragged or rectangular multidimensioned arrays allowed or both?
- Can arrays can be initialized, when they have their storage allocated?
- What kinds of slices are allowed, if any?

Viva Voce on Array

Q1. What do you mean by an Array?

- Array is a set of similar data type.
- Arrays objects store multiple variables with the same type.
- It can hold primitive types and object references.
- Arrays are always fixed

Q2. How to create an Array?

Same as variable, but you need to add [] after the type.

Example: int marks[ ];

Q3. Advantages and disadvantages of Array?

Advantages:

- Arrays can sort multiple elements at a time.
- We can access an element of Array by using an index.

Disadvantages:

1. We have to declare Size of an array in advance. However, we may not know what size we need at the time of array declaration.

2. The array is static structure. It means array size is always fixed, so we cannot increase or decrease memory allocation.

Q4. Can we change the size of an array at run time?

No.

Q5. Can you declare an array without assigning the size of an array?

No.

Q6. What is the default value of Array?

Any new Array is always initialized with a default value as follows

- For byte, short, int, long – default value is zero (0).
- For float, double – default value is 0.0.
- For Boolean – default value is false.
- For object – default value is null.

Q7. Can we declare array size as a negative number?

No.

Q8. When will we get ArrayStoreException?

If anybody tries to insert integer element in String Array, then we will get ArrayStoreException at run time.

Q9. Can we add or delete an element after assigning an array?

No.

Q10. Is there any difference between int[] a and int a[]?

No .

Q11. What are “jagged” arrays in java?

Jagged Arrays are Arrays are containing arrays of different length. Jagged arrays are also known as multidimensional arrays.

Q12. When ArrayIndexOutOfBoundsException occurs?

It is a run time exception.It will occur when the program tries to access invalid index of an array. Index higher than the size of the array or negative index.

Q13. How do we search a specific element in an array?

We can use Arrays.binarySearch() method. This method uses binary search algorithm.

Q14. If you do not initialize an array what will happen?

Array will have default value.

Q15. Can we use Generics with the array?

No.

Q16. Where is the memory allocated for arrays in Java?

On the heap as arrays in Java are objects.

Q17. What is the two-dimensional array?

An array of an array in Java. We can declare them like int p[][] = new int[4][4] which is a matrix of 4×4.

Q18. Do we have 3-dimensional arrays in Java?

Yes, Java supports N-dimensional array.

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: Strong Typing
25. PPL: Coroutines
26. PPL: Exception Handler in C++
27. PPL: OOP in PHP
28. PPL: Character Data Type
29. PPL: Exceptions
30. PPL: Heap based storage management
31. PPL: Primitive Data Type
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++