MCQs on Types of Arrays

1. Which of the following is a **one-dimensional array** declaration?
A) int arr[10];
B) int arr[3][3];
C) int arr[2][2][2];
D) int arr;

---

2. A two-dimensional array in C can be visualized as:
A) A list of variables
B) A table of rows and columns
C) A tree of nodes
D) A linked list

---

3. How many elements are there in int arr[4][3];?
D) 16 C) 9 B) 12 A) 7

---

4. Which declaration creates a 3x2 (3 rows, 2 columns) integer array?
A) int arr[3,2];
B) int arr[3][2];
C) int arr(3)(2);
D) array int[3][2];

---

5. In memory, a 2D array in C is stored in:
A) Column-major order
B) Row-major order
C) Diagonal-major order
D) Compiler-dependent

---

6. What is the correct way to initialize a 2D array?
A) int arr[2][2] = {1,2,3,4};
B) int arr[2][2] = {{1,2},{3,4}};
C) Both A and B
D) None

---

7. If int arr[2][3] = {{1,2,3},{4,5,6}};, what is arr[1][2]?
D) 6 C) 5 B) 3 A) 2

---

8. What is the total size (in bytes) of int arr[3][4]; if sizeof(int)=4?
D) 48 C) 24 B) 16 A) 12

---

9. How to access the element in the 2nd row and 3rd column of arr[3][3]?
A) arr[3][2]
B) arr[2][3]
C) arr[1][2]
D) arr[2][1]

---

10. What will happen if you partially initialize a 2D array?
A) Remaining elements are undefined
B) Remaining elements are initialized to zero
C) Compilation error
D) Only first row is stored

---

11. Which is valid declaration for a 3D array?
A) int arr[2][3][4];
B) int arr[2][3];
C) int arr[];
D) int arr[2,3,4];

---

12. The expression arr[i][j] is equivalent to:
A) *(arr + i + j)
B) *(*(arr + i) + j)
C) *(arr + j + i)
D) None

---

13. The base address of a 2D array arr[4][3] is:
A) Address of arr[0][0]
B) Address of arr[0][1]
C) Address of arr[1][0]
D) Depends on compiler

---

14. In int arr[2][3];, how many rows and columns are there?
A) 2 rows, 3 columns
B) 3 rows, 2 columns
C) 5 elements total
D) None

---

15. What is the address of arr[i][j] in row-major storage?
A) Base + (i * total_columns + j) * sizeof(element)
B) Base + (j * total_rows + i) * sizeof(element)
C) Base + (i + j)
D) Base + (i * j)

---

16. The number of elements in char arr[3][10]; is:
D) 10 C) 3 B) 30 A) 13

---

17. Which of the following is a **jagged array**?
A) An array with irregular row sizes
B) A rectangular 2D array
C) Not supported directly in C
D) Both A and C

---

18. When we declare int arr[2][3] = {{1,2,3},{4,5}};, the missing element becomes:
A) Undefined
B) Zero
C) Garbage
D) None

---

19. Which statement about 2D arrays is **false**?
A) They are arrays of arrays
B) They are stored in contiguous memory
C) You can skip specifying column size when passing to function
D) You can skip specifying row size when passing to function

---

20. int arr[2][2] = {{1,2},{3,4}}; printf("%d", *arr[1]); outputs:
D) 4 C) 3 B) 2 A) 1

---

21. Which of the following is a valid array of strings declaration?
A) char arr[3][20];
B) string arr[3];
C) char *arr[3];
D) Both A and C

---

22. Which access is invalid for a 2D array int a[3][4];?
A) a[0][0]
B) *(a[1] + 2)
C) *a[2]
D) a[3][0]

---

23. What is the difference between arr and &arr in 2D arrays?
A) Both are identical pointers
B) arr → pointer to first row, &arr → pointer to entire array
C) &arr points to first element
D) None

---

24. The number of bytes in double arr[2][3]; if sizeof(double)=8:
D) 48 C) 24 B) 16 A) 6

---

25. What is the output of:

```
int arr[2][3] = {{1,2,3},{4,5,6}};
printf("%d", *(*(arr+1)+2));
```

D) 6 C) 5 B) 4 A) 3

---

26. In C, multi-dimensional arrays are essentially:
A) Arrays of arrays
B) Pointers to arrays
C) Both A and B
D) None

---

27. Which of these initializations is incorrect?
A) int arr[2][2] = {1,2,3,4};

B) int arr[][2] = {1,2,3,4};
C) int arr[2][] = {1,2,3,4};
D) Both B and C

---

28. What is sizeof(arr) if int arr[2][5]; and sizeof(int)=4?
D) 80 C) 40 B) 20 A) 10

---

29. For a 3D array int a[2][3][4];, total elements are:
D) 36 C) 24 B) 12 A) 9

---

30. Which concept correctly describes the structure of a 3D array?
A) Array of 2D arrays
B) Array of linked lists
C) Array of 1D arrays
D) Tree of elements

Code Snippets with MCQs)

Each of the following 20 code snippets tests your understanding of Types of Arrays in C. Predict the correct output and choose the right option (A, B, C, or D).

Q1.

```
#include <stdio.h>
int main() {
   int arr[5] = {1, 2, 3, 4, 5};
   printf("%d", arr[2]);
   return 0;
}
```

What will be the output?

A) 1 B) 2 C) 3 D) 5

Q2.

```
#include <stdio.h>
int main() {
   int arr[2][3] = {{1, 2, 3}, {4, 5, 6}};
   printf("%d", arr[1][1]);
   return 0;
}
```

What will be the output?

A) 1 B) 2 C) 5 D) 6

Q3.

```
#include <stdio.h>
int main() {
   int arr[3][3] = {{1,2,3},{4,5,6},{7,8,9}};
   printf("%d", *(*(arr + 2) + 1));
   return 0;
}
```

What will be the output?

A) 5 B) 7 C) 8 D) 9

Q4.

```
#include <stdio.h>
int main() {
   int arr[2][3] = {{10,20,30},{40,50,60}};
   printf("%d", *(*arr + 4));
   return 0;
}
```

What will be the output?

A) 10 B) 20 C) 50 D) 60

Q5.

```
#include <stdio.h>
int main() {
   int arr[2][2] = {{1,2},{3,4}};
   printf("%d", arr[1][-1]);
   return 0;
}
```

What will be the output?

A) 1 B) 2 C) Undefined Behavior D) Error

Q6.

```
#include <stdio.h>
int main() {
   int arr[3][2] = {{1,2},{3,4},{5,6}};
   printf("%d", *(arr[2]));
   return 0;
}
```

What will be the output?

A) 2 B) 3 C) 5 D) 6

Q7.

```
#include <stdio.h>
int main() {
   int arr[2][3] = {1,2,3,4,5,6};
   printf("%d", *(arr[0] + 5));
   return 0;
}
```

What will be the output?

A) 3 B) 4 C) 5 D) 6

Q8.

```
#include <stdio.h>
int main() {
   int arr[2][3] = {{1,2,3},{4,5,6}};
   printf("%d", *(*(arr+1)+2));
   return 0;
}
```

What will be the output?

A) 3 B) 4 C) 5 D) 6

Q9.

```
#include <stdio.h>
int main() {
   int arr[3][3] = {0};
   printf("%d", arr[1][1]);
   return 0;
}
```

What will be the output?

A) 0 B) 1 C) Garbage Value D) Compilation Error

Q10.

```
#include <stdio.h>
int main() {
   int arr[2][3] = {{1,2},{4,5,6}};
   printf("%d", arr[0][2]);
   return 0;
}
```

What will be the output?

A) 0 B) 2 C) 4 D) 6

Q11.

```
#include <stdio.h>
int main() {
   int arr[2][2] = {{1},{3,4}};
   printf("%d", arr[0][1]);
   return 0;
}
```

What will be the output?

A) 1 B) 0 C) 3 D) Garbage Value

Q12.

```
#include <stdio.h>
int main() {
   int arr[3][3] = {{1,2,3},{4,5,6},{7,8,9}};
   printf("%d", *(arr[1] + 2));
   return 0;
}
```

What will be the output?

A) 5 B) 6 C) 7 D) 8

Q13.

```
#include <stdio.h>
int main() {
   int arr[3][3] = {{1,2,3},{4,5,6},{7,8,9}};
   printf("%d", **arr);
   return 0;
}
```

What will be the output?

A) 1 B) 2 C) 4 D) 7

Q14.

```
#include <stdio.h>
int main() {
   int arr[2][3] = {{10,20,30},{40,50,60}};
   printf("%d", arr[1][0] + arr[0][2]);
   return 0;
}
```

What will be the output?

A) 50 B) 70 C) 90 D) 100

Q15.

```
#include <stdio.h>
int main() {
   int arr[2][3] = {{1,2,3},{4,5,6}};
   printf("%d", sizeof(arr)/sizeof(arr[0][0]));
   return 0;
}
```

What will be the output?

A) 6 B) 12 C) 24 D) 8

Q16.

```
#include <stdio.h>
int main() {
   int arr[2][3] = {{1,2,3},{4,5,6}};
   printf("%d", *(arr[0]+1) + *(arr[1]+2));
   return 0;
}
```

What will be the output?

A) 7 B) 8 C) 9 D) 10

Q17.

```
#include <stdio.h>
int main() {
   int arr[3][3] = {{1,2,3},{4,5,6},{7,8,9}};
   int *p = &arr[0][0];
   printf("%d", *(p + 7));
   return 0;
}
```

What will be the output?

A) 5 B) 7 C) 8 D) 9

Q18.

```
#include <stdio.h>
int main() {
   int arr[2][2] = {{10,20},{30,40}};
   printf("%p %p", arr, arr+1);
   return 0;
}
```

What will be the output?

A) Same address B) arr+1 jumps by one integer C) arr+1 jumps by one row D) Undefined

Q19.

```
#include <stdio.h>
int main() {
   int arr[2][3] = {{1,2,3},{4,5,6}};
   printf("%d", *(*arr + 3));
   return 0;
}
```

What will be the output?

A) 3 B) 4 C) 5 D) 6

Q20.

```
#include <stdio.h>
int main() {
   int arr[3][3] = {{1,2,3},{4,5,6},{7,8,9}};
   printf("%d", *(*(arr+2)+0));
   return 0;
}
```

What will be the output?

A) 6 B) 7 C) 8 D) 9

String MCQs

Q1. What is printed?

```
char s[] = "Hello";
printf("%c", s[1]);
```

A) H
B) e
C) l
D) o

Q2. Output?

```
char s[10];
strcpy(s, "Cat");
printf("%s", s);
```

A) Ca
B) Cat
C) C
D) Error

Q3. What will this print?

```
char s[10] = "ABC";
s[1] = 'Z';
printf("%s", s);
```

A) ABC
B) AZC
C) ZBC
D) ABZ

Q4. What is the length?

```
char s[] = "Test";
printf("%d", strlen(s));
```


Dr.Kavitha N
Dept. of IT

A) 3
B) 4
C) 5
D) 6

Q5. Output?

```
char s1[10]="Hi", s2[]="All";
strcat(s1, s2);
printf("%s", s1);
```

A) Hi
B) All
C) HiAll
D) Error

Q6. What happens?

```
char s[5] = "Hello";
printf("%s", s);
```

A) Prints Hello
B) Compiles but overflow
C) Error
D) Prints Hell

Q7. Output?

```
char s[20];
gets(s);
printf("%s", s);
```

A) Reads one word
B) Reads whole line
C) Reads only chars
D) Error

Q8. What is printed?

```
char s[20] = "World";
```


Dr.Kavitha N
Dept. of IT

printf("%c", s[10]);

A) W
B) Space
C) '\0'
D) Garbage

Q9. strcmp result?

printf("%d", strcmp("A", "B"));

A) 0
B) Positive
C) Negative
D) Error

Q10. What is substring printed?

char s[]="Laptop";
printf("%c%c", s[2], s[3]);

A) La
B) ap
C) pt
D) to

Q11. Output?

char s1[]="ABC", s2[]="ABC";
printf("%d", strcmp(s1,s2));

A) 0
B) 1
C) -1
D) Error

Q12. Output?

char s[]="Hello";
s[0] = '\0';

Dr.Kavitha N
Dept. of IT

```
printf("%s", s);
```

A) Hello
B) ello
C) (empty string)
D) Error

Q13. What is printed?

```
char s1[10] = "Hi";
char s2[] = "There";
strncat(s1, s2, 2);
printf("%s", s1);
```

A) Hi
B) HiT
C) HiTh
D) HiThere

Q14. Size of string?

```
char s[] = {'A','B','C','\0'};
printf("%d", strlen(s));
```

A) 2
B) 3
C) 4
D) 1

Q15. Output?

```
char s1[]="Code";
printf("%c", s1[strlen(s1)-1]);
```

A) C
B) o
C) d
D) e

Q16. strcmp comparison?

Dr.Kavitha N
Dept. of IT

strcmp("apple", "app");

A) 0
B) Negative
C) Positive
D) Error

Q17. What happens?

char s[5];
strcpy(s, "Laptop");
printf("%s", s);

A) Prints Laptop
B) Overflow occurs
C) Error
D) Prints Lap

Q18. Output?

char s[]="abcd";
printf("%c", s[strlen(s)]);

A) a
B) d
C) '\0'
D) Error

Q19. Result of concatenation?

char s[20]="A";
strcat(s, "B");
printf("%s", s);

A) A
B) AB
C) B
D) Error

Q20. Output?

Dr.Kavitha N
Dept. of IT

```
char s[]="Test";
printf("%d", sizeof(s));
```

A) 3
B) 4
C) 5
D) Depends on OS

Dr.Kavitha N
Dept. of IT