Arrays Interview Questions And Answers Guide.

Global Guideline.

https://globalguideline.com/

Arrays Job Interview Preparation Guide.

Question # 1

What will be output if you will execute following c code?

```
#include<stdio.h>
void main(){
    char arr[7]="Network";
    printf("%s",arr);
```

Answer:-

Garbage value.

Read More Answers.

Question # 2

What will be output if you will execute following c code?

```
#include<stdio.h>
void main(){
    char arr[11]="The African Queen";
    printf("%s",arr);
```

Answer:-

null.

Read More Answers.

Question # 3

What will be output if you will execute following c code?

```
#include<stdio.h>
void main(){
    char arr[20]="MysticRiver";
    printf("%d",sizeof(arr));
```

Answer:-

20

Read More Answers.

Question # 4

What will be output if you will execute following c code?

```
#include<stdio.h>
void main(){
    int const SIZE=5;
    int expr;
    double value[SIZE]={2.0,4.0,6.0,8.0,10.0};
    expr=1|2|3|4;
    printf("%f",value[expr]);
```

Answer:-

Compilation error

Read More Answers.

Question # 5

What will be output if you will execute following c code?

```
#include<stdio.h>
enum power{
    Dalai,
    Vladimir=3,
    Barack,
    Hillary
};
void main(){
    float leader[Dalai+Hillary]={1.f,2.f,3.f,4.f,5.f};
    enum power p=Barack;
    printf("%0.f",leader[p>>1+1]);
```

Answer:-

2

Read More Answers.

Question # 6

What will be output if you will execute following c code?

```
#include<stdio.h>
#define var 3
void main(){
    char *cricket[var+~0]={"clarke","kallis"};
    char *ptr=cricket[1+~0];
    printf("%c",*++ptr);
```

Answer:-

l

Read More Answers.

Question # 7

What will be output if you will execute following c code?

```
#include<stdio.h>
#define var 3
void main(){
    char data[2][3][2]={0,1,2,3,4,5,6,7,8,9,10,11};
    printf("%o",data[0][2][1]);
```

Answer:-

5

Read More Answers.

Question # 8

What will be output if you will execute following c code?

```
#include<stdio.h>
#define var 3
void main(){
    short num[3][2]={3,6,9,12,15,18};
    printf("%d   %d",*(num+1)[1],**(num+2));
```

Answer:-

15 15

Read More Answers.

Question # 9

What will be output if you will execute following c code?

```
#include<stdio.h>
#define var 3
void main(){
    char *ptr="cquestionbank";
    printf("%d",-3[ptr]);
```

Answer:-

-101

Read More Answers.

Question # 10

What will be output if you will execute following c code?

```
#include<stdio.h>
void main(){
    long  myarr[2][4]={0l,1l,2l,3l,4l,5l,6l,7l};
    printf("%ldt",myarr[1][2]);
    printf("%ld%ldt",*(myarr[1]+3),3[myarr[1]]);
    printf("%ld%ld%ldt" ,*(*(myarr+1)+2),*(1[myarr]+2),3[1[myarr]]);
```

Answer:-

6   77   667

Read More Answers.

Question # 11

What will be output if you will execute following c code?

```
#include<stdio.h>
void main(){
    int array[2][3]={5,10,15,20,25,30};
    int (*ptr)[2][3]=&array;
    printf("%dt",***ptr);
    printf("%dt",***(ptr+1));
    printf("%dt",**(*ptr+1));
    printf("%dt",*(*(*ptr+1)+2));
```

Answer:-

5   Garbage value   20   30

Read More Answers.

Question # 12

What will be output if you will execute following c code?

```
#include<stdio.h>
void main(){
    static int a=2,b=4,c=8;
    static int *arr1[2]={&a,&b};
    static int *arr2[2]={&b,&c};
    int* (*arr[2])[2]={&arr1,&arr2};
    printf("%d %dt",*(*arr[0])[1],  *(*(**(arr+1)+1)));
```

Answer:-

4 8

Read More Answers.

Question # 13

What will be output if you will execute following c code?

```
#include<stdio.h>
#include<math.h>
double myfun(double);
void main(){
    double(*array[3])(double);
    array[0]=exp;
    array[1]=sqrt;
    array[2]=myfun;   printf("%.1ft",(*array)((*array[2])((**(array+1))(4))));

}
double myfun(double d){
       d-=1;
       return d;
}
```

Answer:-

2.7

Read More Answers.

Question # 14

What will be output if you will execute following c code?

```
#include<stdio.h>
#include<math.h>
typedef struct{
    char *name;
    double salary;
}job;
void main(){
    static job a={"TCS",15000.0};
    static job b={"IBM",25000.0};
    static job c={"Google",35000.0};
    int x=5;
    job * arr[3]={&a,&b,&c};
    printf("%s  %ft",(3,x>>5-4)[*arr]);
}
double myfun(double d){
       d-=1;
       return d;
```

Answer:-

Google 35000.000000

Read More Answers.

Question # 15

What will be output if you will execute following c code?

```
#include<stdio.h>
union group{
    char xarr[2][2];
    char yarr[4];
};
void main(){
    union group x={'A','B','C','D'};
    printf("%c",x.xarr[x.yarr[2]-67][x.yarr[3]-67]);
}
```

Answer:-

B

Read More Answers.

Question # 16

What will be output if you will execute following c code?

```
#include<stdio.h>
void main(){
    int a=5,b=10,c=15;
    int *arr[3]={&a,&b,&c};
    printf("%d",*arr[*arr[1]-8]);
```

Answer:-

Compilation error

Read More Answers.

Question # 17

What will be output if you will execute following c code?

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

Answer:-

18 0 5

Read More Answers.

Question # 18

What will be output if you will execute following c code?

```
#include<stdio.h>
void main(){
    int xxx[10]={5};
    printf("%d %d",xxx[1],xxx[9]);
```

Answer:-

0 0

Read More Answers.

Question # 19

What will be output if you will execute following c code?

```
#include<stdio.h>
#define WWW -1
enum {cat,rat};
void main(){
    int Dhoni[]={2,'b',0x3,01001,'x1d','111',rat,WWW};
    int i;
    for(i=0;i<8;i++)
        printf(" %d",Dhoni[i]);
```

Answer:-

2 98 3 513 29 73 1 -1

Read More Answers.

Question # 20

What will be output if you will execute following c code?

```
#include<stdio.h>
void main(){
    long double a;
    signed char b;
    int arr[sizeof(!a+b)];
    printf("%d",sizeof(arr))
}
```

Answer:-

4

Read More Answers.

Question # 21

Tell me do array subscripts always start with zero?

Answer:-

Yes. If you have an array a[MAX] (in which MAX is some value known at compile time), the first element is a[0], and the last element is a[MAX-1]. This arrangement is different from what you would find in some other languages. In some languages, such as some versions of BASIC, the elements would be a[1] through a[MAX], and in other languages, such as Pascal, you can have it either way.

Read More Answers.

Question # 22

Explain is it valid to address one element beyond the end of an array?

Answer:-

It's valid to address it, but not to see what's there. (The really short answer is, "Yes, so don't worry about it.") With most compilers, if you say

```
int i, a[MAX], j;
```

then either i or j is at the part of memory just after the last element of the array. The way to see whether i or j follows the array is to compare their addresses with that of the element following the array. The way to say this in C is that either

```
& i == & a[ MAX ]
```

is true or

```
& a[ MAX ] == & j
```

is true. This isn't guaranteed; it's just the way it usually works.

Read More Answers.

Question # 23

Tell me can the size of operator be used to tell the size of an array passed to a function?

Answer:-

No. There's no way to tell, at runtime, how many elements are in an array parameter just by looking at the array parameter itself. Remember, passing an array to a function is exactly the same as passing a pointer to the first element. This is a Good Thing. It means that passing pointers and arrays to C functions is very efficient.
It also means that the programmer must use some mechanism to tell how big such an array is. There are two common ways to do that. The first method is to pass a count along with the array. This is what memcpy() does, for example:

```
char    source[ MAX ], dest[ MAX ];
/* ... */
memcpy( dest, source, MAX );
```

Read More Answers.

Question # 24

Tell me is it better to use a pointer to navigate an array of values, or is it better to use a subscripted array name?

Answer:-

It's easier for a C compiler to generate good code for pointers than for subscripts.
Say that you have this:

```
/* X is some type */
X       a[ MAX ];       /* array */
X       *p;     /* pointer */
X       x;      /* element */
int     i;      /* index */
```

Here's one way to loop through all elements:

```
/* version (a) */
for ( i = 0; i < MAX; ++i )
{
        x = a[ i ];
        /* do something with x */
}
```

On the other hand, you could write the loop this way:

```
/* version (b) */
for ( p = a; p < & a[ MAX ]; ++p )
{
        x = *p;
        /* do something with x */
}
```

Read More Answers.

Question # 25

Explain can you assign a different address to an array tag?

Answer:-

No, although in one common special case, it looks as if you can. An array tag is not something you can put on the left side of an assignment operator. (It's not an "lvalue," let alone a "modifiable lvalue.") An array is an object; the array tag is a pointer to the first element in that object.
For an external or static array, the array tag is a constant value known at link time. You can no more change the value of such an array tag than you can change the value of 7.
Assigning to an array tag would be missing the point. An array tag is not a pointer. A pointer says, "Here's one element; there might be others before or after it." An array tag says, "Here's the first element of an array; there's nothing before it, and you should use an index to find anything after it." If you want a pointer, use a pointer.

Read More Answers.

Question # 26

Can you please explain the difference between array_name and &array_name?

Answer:-

One is a pointer to the first element in the array; the other is a pointer to the array as a whole.
An array is a type. It has a base type (what it's an array of), a size (unless it's an "incomplete" array), and a value (the value of the whole array). You can get a pointer to this value:

```
char a[ MAX ];      /* array of MAX characters */
char *p;            /* pointer to one character */
/* pa is declared below */
pa = & a;
p = a;  /* = & a[ 0 ] */
```

After running that code fragment, you might find that p and pa would be printed as the same value; they both point to the same address. They point to different types of MAX characters.
The wrong answer is

```
char *( ap[ MAX ] );
```

which is the same as this:

```
char *ap[ MAX ];
```

This code reads, "ap is an array of MAX pointers to characters."

Read More Answers.

Question # 27

Tell me why can't constant values be used to define an array's initial size

Answer:-

There are times when constant values can be used and there are times when they can't. A C program can use what C considers to be constant expressions, but not everything C++ would accept.
When defining the size of an array, you need to use a constant expression. A constant expression will always have the same value, no matter what happens at runtime, and it's easy for the compiler to figure out what that value is. It might be a simple numeric literal:

```
char a[ 512 ];
```

Or it might be a "manifest constant" defined by the preprocessor:

```
#define MAX     512
/* ... */
char    a[ MAX ];
```

Or it might be a sizeof:

```
char a[ sizeof( struct cacheObject ) ];
```

Or it might be an expression built up of constant expressions:

```
char buf[ sizeof( struct cacheObject ) * MAX ];
```

Enumerations are allowed too.
An initialized const int variable is not a constant expression in C:

```
int max = 512; /* not a constant expression in C */
char buffer[ max ]; /* not valid C */
```

Using const ints as array sizes is perfectly legal in C++; it's even recommended. That puts a burden on C++ compilers (to keep track of the values of const int variables) that C compilers don't need to worry about. On the other hand, it frees C++ programs from using the C preprocessor quite so much.

Read More Answers.

Question # 28

Can you please explain the difference between string and an array?

Answer:-

An array is an array of anything. A string is a specific kind of an array with a well-known convention to determine its length.
There are two kinds of programming languages: those in which a string is just an array of characters, and those in which it's a special type. In C, a string is just an array of characters (type char), with one wrinkle: a C string always ends with a NUL character. The "value" of an array is the same as the address of (or a pointer to) the first element; so, frequently, a C string and a pointer to char are used to mean the same thing.
An array can be any length. If it's passed to a function, there's no way the function can tell how long the array is supposed to be, unless some convention is used. The convention for strings is NUL termination; the last character is an ASCII NUL (") character.
String literals are arrays of characters (type char), not arrays of constant characters (type const char). The ANSI C committee could have redefined them to be arrays of const char, but millions of lines of code would have screamed in terror and suddenly not compiled. The compiler won't stop you from trying to modify the contents of a string literal. You shouldn't do it, though. A compiler can choose to put string literals in some part of memory that can't be modified-in ROM, or somewhere the memory mapping registers will forbid writes. Even if string literals are someplace where they could be modified, the compiler can make them shared.

Read More Answers.

Question # 29

Explain Array of pointers?

Answer:-

An array of pointers is an array consisting of pointers. Here, each pointer points to a row of the matrix or an element. E.g char *array [] = {â€œaâ€•, â€œbâ€•}. This is an array of pointers to to characters.

Read More Answers.

Question # 30

What is two-dimensional array?

Answer:-

An array with two dimensions is called as a two-dimensional array. It is also called as a matrix. In C, a two dimensional array is initialized as int arr[nb_of_rows][nb_of_columns]. Hence, two dimensional arrays can be considered as a grid. An element in a two dimensional can be accessed by mentioning its row and column. If the array has 20 integer values, it will occupy 80 bytes in memory (assuming 4 bytes for an integer). All of the bytes are in consecutive memory locations, the first row occupying the first 20 bytes, the second the next 20, and so on.

Read More Answers.

Question # 31

What is Array?

Answer:-

An array is a series of elements. These elements are of the same type. Each element can be individually accessed using an index. For e.g an array of integers. Array elements are stored one after another (contiguous) in the memory. An array can have more than one dimension. First element in an array starts with 0.

Read More Answers.

Data Structure Most Popular Interview Topics.

1 : Sort And Searching Frequently Asked Interview Questions and Answers Guide.

2 : Linked list Frequently Asked Interview Questions and Answers Guide.

3 : Data Architect Frequently Asked Interview Questions and Answers Guide.

4 : Data Center Manager Frequently Asked Interview Questions and Answers Guide.

About Global Guideline.

Global Guideline is a platform to develop your own skills with thousands of job interview questions and web tutorials for fresher's and experienced candidates. These interview questions and web tutorials will help you strengthen your technical skills, prepare for the interviews and quickly revise the concepts. Global Guideline invite you to unlock your potentials with thousands of **Interview Questions with Answers** and much more. Learn the most common technologies at Global Guideline. We will help you to explore the resources of the World Wide Web and develop your own skills from the basics to the advanced. Here you will learn anything quite easily and you will really enjoy while learning. Global Guideline will help you to become a professional and Expert, well prepared for the future.

* This PDF was generated from https://GlobalGuideline.com at **November 29th, 2023**

* If any answer or question is incorrect or inappropriate or you have correct answer or you found any problem in this document then don't hesitate feel free and e-mail us we will fix it.

You can follow us on FaceBook for latest Jobs, Updates and other interviews material.
www.facebook.com/InterviewQuestionsAnswers

Follow us on Twitter for latest Jobs and interview preparation guides
https://twitter.com/InterviewGuide

Best Of Luck.

Global Guideline Team
https://GlobalGuideline.com
Info@globalguideline.com