**C Interview questions**

**<Freshers – 15 Years>**

1. Write a program to print the sum of digits of any number

Input: 16754

Output: 23

1. Write a program to convert a binary number to a decimal number

Input: 1011

Output: 11

1. Write a program to find products of digits of any number

Input: 246

Output: 48

1. Write a program to generate fibonacci series

Output: 0,1,1,2,3,5,8,13,34,55,89…

1. Write a program to print armstrong numbers from 100 to 999

**Output**: 0, 1, 153, 370, 371 and 407

1. Please choose right one for these statements <bold one is ANS>

By the C compiler – Function **Declaration**/Definition/Call?

By the Programmer – Function Declaration/**Definition**/Call?

By the operating system – Function Declaration/Definition/**Call**?

1. Write a program to check the entered number is Palindrome or not

**Output**: Entered num and revered num should be same like 6556



if entered num is 32 output is 10 0000

if entered num is 31 output is 01 1111

1. Write a program to search an element through binary search
2. Write a program to print number of positive, negative and zero numbers in an array

#include <stdio.h>

int main()

{

int arr[]={0,1,2,4,0,3,1,0,-1,2,0,0,0,0};

int n=sizeof(arr)/sizeof(int), i, positive = 0, negative = 0, zero = 0;

for (i = 0; i < n; i++)

{

if (arr[i] > 0)

positive++;

else if (arr[i] < 0)

negative++;

else

zero++;

}

printf("Number of positive elements: %d\n", positive);

printf("Number of negative elements: %d\n", negative);

printf("Number of zeros: %d", zero);

return 0;

}

1. Can you interpret below pointer expressions?

a = \*ptr++;

**//{a=\*ptr; ptr=ptr+1;}**

a = \* ++ptr;

**//{ptr=ptr+1; a=\*ptr;}**

a = ++\*ptr;

**//{\*ptr=\*ptr+1; a=\*ptr;}**

a = (\*ptr)++;

**//{a=\*ptr; \*ptr=\*ptr+1;}**

1. Below is the array defined with the name **arr**

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| 5 | 6 | 7 | 8 |
| 9 | 10 | 11 | 12 |

Can you give me dereference expression to get value **12**?

ANS: \*(\*(arr+2) +3)

Can you give me dereference expression to get address of **9**?

ANS: Points to 0th element of ith <2 here> 1-D array. i.e **\*(arr + 2)**

1. Can you elaborate these expressions?

\*(\*(arr+i)+j)+k : **Points to Kth element of Jth 1-D array of I th 2-D array**

\*(\*(\*(arr +i) +j) +k) : **Gives the value of K th element of J th 1-D array of I th 2-D array**

1. Will below program work fine?

int i=2, \*ip=&i;

float f =1.3, \*fp = &f;

ip =fp;

//if above is wrong, how to fix it?

void \*vp;

vp =ip;

vp = fp;

1. Can you write all dynamic memory declarations for malloc, calloc, realloc?

void \*malloc(size\_t size);

void \*calloc(size\_t n, size\_t size);

void \*realloc(void \*ptr, size\_t newsize);

void free(void \*ptr);

1. Can you explain how free() works and what happen if we use free twice?

When you use malloc you are telling the PC that you want to reserve some memory location on the heap just for you. The computer gives back a pointer to the first byte of the addressed space.

When you use free you are actually telling the computer that you don't need that space anymore, so it marks that space as available for other data.

The pointer still points to that memory address. At this point that same space in the heap can be returned by another malloc call.

When you invoke free a second time, you are not freeing the previous data, but the new data, and this may crash the program

1. Can you explain why free not taking number of bytes to free instead of ptr?
2. How does free() know the size of memory to be deallocated?

If you allocate 6 bytes of memory, the system actually reserves 10.

The first 4 contains the amount of data you requested (6 bytes) and then the return value of the malloc is a pointer to the first byte of unused data in allocated 10 bytes

When you call free on this pointer, the system will look up 4 bytes backwards to know that it originally allocated 10 bytes

so that it knows how much to free. This system prevents you from providing the amount of data to free as an extra parameter to free itself.

1. How to call a function through function pointer?

int (\* fp) (int, int);

int add (int, int);

fp = add//assigning address of function add() to pointer fp

(\* fp)(5,10);

1. Declare an array of 4 function pointers which takes two args with data type int and return int

Int (\*fp[4] )(int, int);

1. Write pointer versions of strcmp, strcpy, strcat

**Strcmp:**

int pstrcmp(char \*str1, char \*str2){

while(\*str1!=’\0’ && \*str2!=’\0’ && \*str1 == \*str2){

str1++;str2++;

}

If(\*str1 == \*str2)

Return 0;

Else

Return (\*str1 - \*str2);

}

**Strcpy:**

char \*pstrcpy(char \*str1, char \*str2){

while(\*str2 != ‘\0’){

\*str1 = \*str2;

str1++;

str2++;

}

\*str1 = ‘\0’;

Return str1;

}

//you can ask much better version than above also

**Strcat:**

char \* pstrcat(char \*str1, char \*str2){

while(\*str1 != ’\0’)

Str1++;

while(\*str2 != ’\0’){

\*str1 = \*str2;

str1++;str2++;

}

\*str1 = ‘\0’;

return str1;

}

1. Is below code proper to use directly? If not why?

struct {

char name[10];

int data;

};

//trying to know better on struct tagname and variable creation part

1. Write a program to know the size of struct without size of operator

#include<stdio.h>

struct student{

int roll;

char name[10];

float marks;

};

int main(){

struct student \*ptr = 0;

ptr++;

printf("Size of the structure student: %p\n",ptr);

return 0;

}

//Size of the structure student: 0x14 = 20

1. Write a C program to display its own source code as output

#include <stdio.h>

int main() {

FILE \*fp;

int c;

// open the current input file

fp = fopen(\_\_FILE\_\_,"r");

do {

c = getc(fp); // read character

putchar(c); // display character

}while(c != EOF); // loop until the end of file is reached

fclose(fp);

return 0;

}

1. Write a program to reverse words in a string

#include <stdio.h>

void reverse(char\* begin, char\* end)

{

char temp;

while (begin < end) {

temp = \*begin;

\*begin++ = \*end;

\*end-- = temp;

}

}

// Function to reverse words\*/

void reverseWords(char\* s)

{

char\* word\_begin = s;

// Word boundary

char\* temp = s;

// Reversing individual words as

// explained in the first step

while (\*temp) {

temp++;

if (\*temp == '\0') {

reverse(word\_begin, temp - 1);

}

else if (\*temp == ' ') {

reverse(word\_begin, temp - 1);

word\_begin = temp + 1;

}

}

// Reverse the entire string

reverse(s, temp - 1);

}

int main()

{

char s[] = "i am a embedded software developer";

char\* temp = s;

reverseWords(s);

printf("%s", s);

return 0;

}

1. Give o/p for below program and explain how it will work?

#include <stdio.h>

typedef char\* ptr;

#define PTR char\*

int main()

{

ptr a, b, c;

PTR x, y, z;

printf("sizeof a:%zu\n" ,sizeof(a) );

printf("sizeof b:%zu\n" ,sizeof(b) );

printf("sizeof c:%zu\n" ,sizeof(c) );

printf("sizeof x:%zu\n" ,sizeof(x) );

printf("sizeof y:%zu\n" ,sizeof(y) );

printf("sizeof z:%zu\n" ,sizeof(z) );

return 0;

}

DS:

1. Explain stack and queue in data structure with examples
2. **Stack**: LIFO; Example<plates> you can only take a plate from the top of the stack, and you can only add a plate to the top of the stack

**Queue**: FIFO; Example is a single-lane one-way road, where the vehicle enters first, exits first

1. Write a program to implement a Queue using two Stacks
2. Why Linked list is used?

A linked list consists of a sequence of nodes, each containing a value and a pointer to the next node in the list. This makes linked lists more flexible than arrays, as they can be easily resized and modified without needing to move large amounts of data around

1. Write a C Program to insert a Node in a Linked List
2. Write a C Program to delete a Node in a Linked List
3. Write a C Program to insert a Node in the beginning in a Linked List
4. Write a C Program to reverse the nodes in a Linked List
5. Write a C Program to identify the mid node in a Linked List
6. Write a C Program to find the loop in a Linked List
7. Write a C Program to SWAP alternat nodes in a Linked List