1. What is the maximum number of arguments that can be passed in a single function?

**->253**

int main()

{

int i,j;

printf("enter value");

scanf("%d",&++i);// if enter 4 then output also 4(no increment).

printf("enter value");

scanf(“%d”,&j+4);// if entre any integer then output is 0(zero).

//**cin>>j+4//is also error in C++**

printf("%d",i);

return(0);

}

In C, If the **array** is Local variable ( declared inside a function) , then it is unknown as it will take garbage **value**. If it is declared Global or static, then it will be initialized to 0 by**default** .In java, it is Zero(0).

{int a[]={1.5, 24.6, 3.7, 4.4, 5.5};

int length = sizeof(a)/sizeof(a[0]); //print length=5

printf(" lenght/size is %d", sizeof(a));// print 20

return(0);

}

//Program: Reverese Array itself

there is need to store the size of array(n) in the another variable

otherwise value will change of original variable. See following program.

***len=n; // store another variable***

*for(i=0; i<len/2;i++)*

*{*

*temp=a[len-1];*

*a[len-1]=a[i];*

*a[i]=temp;*

*len--;*

*}*

->two **pointer** variables **can point** to the **same** object:

**you can** store int type **variables**, like

int x = 4;

int y = 4;

same way **you can** store address of a **variable** in two **pointers** like

int \*p = &x;

int \*q = &x.

->Yes, a declared **pointer has its own** location in memory.(has its own memory location).But **void pointer** points nothing

3. How much memory does a pointer variable take?

->depends on the **address width** of that computer. If you have a 32-bit processor, a **pointer** will occupy 32 bits or 4 bytes. If you have a 64 bit computer, it will occupy 64 bits or 8 bytes.

4.How many bytes is a double pointer?

If you're on a **32**-bit system, probably all the pointer types will be **4 bytes** and double will be **8 bytes.**

5. Can a reference be null?

-> no. so reference cant be change or null whereas pointer can be change .

a declared **pointer has its own** location in memory.

6. **Program-1**

1. #include <stdio.h>
2. int main()
3. {
4. int \*ptr, a = 10;
5. ptr = &a;
6. \*ptr += 1;
7. printf("%d,%d/n", \*ptr, a);
8. }

Output: 11 11\

**Program-2**

int main()

{ char \*ptr;

char str[]="akashthakare";

**ptr=str;**

printf("%c",\*ptr);// print ‘a’;

printf("%c",ptr[4]);// print ‘h’;

**\*ptr=str[4];** // don’t need ‘&’ bz we already pointer pointed str.

printf("%c",\*ptr);//print ‘h’;

}

**Program-3**

**int main()**

{

char \*ptr;

char str[]="akashthakare";

**ptr=str; // first need it then i=3, ptr++, ptr+=3, str[i+4] etc..**

int i=3;

**\*ptr=str[i]; OR ptr=&str[i];**

printf("%c",\*ptr); // print ‘s’;

}

**Program-4**

int main

{ char \*ptr;

char str[]="akashthakare";

**ptr=str;// first need**

ptr++; **Or** ptr+=3 print ‘s’.

printf("%c",\*ptr);// print ‘k’;

}

**Program-5**

{ char \*ptr;

char str[]="akashthakare";

ptr=str;

int i=3;

**1.\*ptr=str[++i];// print ‘h’**

**2.\*ptr=str[i++];// print ‘s’**

**3.** **\*ptr=str[--i];// print ‘a’**

**4.** **\*ptr=str[i--];// print ‘s’**

**5.** **\*ptr=str[i+4];//print ‘a’**

printf("%c",\*ptr);

}

**Program-6**

int main()

{ char \*ptr1;

int \*ptr2;

int x=123;

char str[]="akashthakare";

ptr1=str; // ptr1=&str// error, already point first location.

ptr2=&x; // ptr2=x // error, we need address so use '&x'.

printf("%c\n",\*ptr1);// 'a'

printf("%d",\*ptr2);//'123'

printf("%d\n",\*ptr1);// 97(ASCII Code)

printf("%d",\*ptr2);//'123'

printf("%d\n",ptr1);//6487584

printf("%d",ptr2);//6487612

}

#### 7.Void Pointer In C?

* Void pointer is a generic pointer that can be used to point another variable of **any data type.**
* Void pointer can store the address of variable belonging to any of the data type. So, void pointer is also called as general purpose pointer.

**Note:**  
**int** pointer can be used to point a variable of int data type and **char** pointer can be used to point a variable of char data type.

#### 8. ****What Is Static Function In C?****

All functions are global(extern) by default in a C program/file. But, static keyword makes a function as **a local function** which can be accessed only by the program/file where it is declared and defined. Other programs/files **can’t** **access** these static functions.

**Static vs Extern**

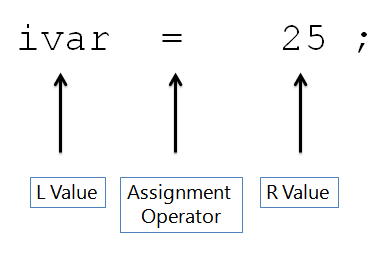
9. **Null** **pointer** is a pointer which is pointing to nothing. Null pointer points to **empty** location in memory. Value of null pointer is **0**. We can make a pointer to point to null as below.

int\*p=NULL;  
char \*p = NULL;

10. **stdin , stdout , stderr** ( standard streams)

When a **C program** starts its execution the **program** automatically opens three standard **streams** named stdin , stdout , and **stderr** . These are attached for every**C program**. The first standard **stream** is used for input buffering and the other two are used for output. These **streams** are sequences of bytes.

1. 11. L-Value R-value.



L Value must be any **Variable**. L Value **Cannot be Constant**, **Function** , **data type** in C. Also cant be MACRO,ENUM

#define MAX 20

**int** main()

{

MAX = 20; //Error i.e 20=20 not possible

**return**(0);

}

**enum** {JAN,FEB,MARCH};

**int** main()

{

JAN = 20; //Error

**return**(0);

}

Also can’t be data type

**struct** book{

**char** \*name;

**int** pages;

};

**void** main()

{

book = {"C Programming",100};// book is data type

}

11. Trailing Arguments i.e Default **arguments**. Allows a function to be called without providing one or more **trailing arguments**.