## Loops and Pointers

#include <stdio.h>

int main () {

int value\_array[] = {42, 366, 458, 112};

int a, \*point[4];

printf("TechVidvan Tutorial: Array of pointers!\n\n");

for (a = 0; a < 4; a++) {

point[a] = &value\_array[a]; // assigning address!

}

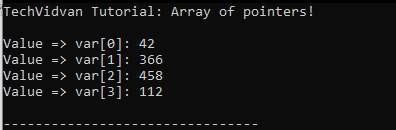
for (a= 0; a < 4; a++) {

printf("Value => var[%d]: %d\n", a, \*point[a] );

}

return 0;

}



#include <stdio.h>

int main () {

int val;

int \*pt;

int \*\*point;

val = 55;

pt = &val;

point = &pt;

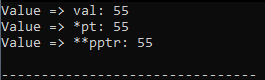
printf("Value => val: %d\n", val);

printf("Value => \*pt: %d\n", \*pt );

printf("Value => \*\*pptr: %d\n", \*\*point);

return 0;

}



#include <stdio.h>

#include <conio.h>

void main()

{

int a[6] = {10, 20, 30, 40, 50, 60};

int \*p;

int i;

p = a;

for (i = 0; i < 6; i++)

{

printf("%d ", \*p); // value of elements of array

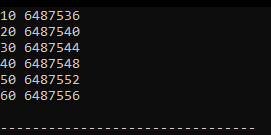
printf("%u\n", p); // Address of array

p++; // Move to the next element in the array

}

getch(); // Wait for a key press before exiting

}



#include<stdio.h>

#include <conio.h>

void main()

{

int a[10];

int i,sum=0;

int \*ptr;

printf("Enter 10 elements:n");

for(i=0;i<10;i++)

scanf("%d",&a[i]);

ptr = a; /\* a=&a[0] \*/

for(i=0;i<10;i++)

{

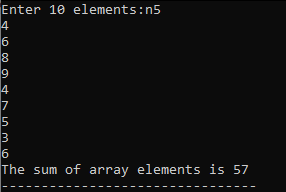
sum = sum + \*ptr;

ptr++;

}

printf("The sum of array elements is %d",sum);

}



#include<stdio.h>

int main() {

int i, j, temp1, temp2;

int arr[8] = {5, 3, 0, 2, 12, 1, 33, 2};

int \*ptr;

// Bubble sort algorithm

for (i = 0; i < 7; i++) {

for (j = 0; j < 7 - i; j++) {

if (\*(arr + j) > \*(arr + j + 1)) {

ptr = arr + j;

temp1 = \*ptr++;

temp2 = \*ptr;

\*ptr-- = temp1;

\*ptr = temp2;

}

}}

for (i = 0; i < 8; i++) {

printf(" %d", arr[i]);

}

return 0;

}



GATE QUESTIONS :

1)

struct node

{

inti;

float j;

};

struct node \*s[10];

The above C declaration defines

GATE CSE 2000

1. An array, each element of which is pointer to a structure of type node

2. A structure of 2 fields, each field being a pointer to an array of 10 elements

3. A structure of 3 fields: an integer, a float, and an array of 10 elements

4. An array, each element of which is a structure of type node

SOLUTION:

The correct option is 1.

The program indicates that the list s is created with 10 elements where it stores the address(hence become pointer) of a structure of type node .

2)

Assume that objects of the type short, float and long occupy 2 bytes, 4 bytes and 8 bytes,

respectively. The memory requirement for variable t, ignoring alignment

GATE CSE 2000

struct {

short s [5];

union {

float y;

long z;

}u;

} t;

1. 22 bytes

2. 18bytes

3. 14 bytes

4. 10 bytes

SOLUTION :

The correct option is 2 .

As type short occupy 2 bytes , the structure occupies 5\*2=10 bytes.

In case of union , larger size type between long and float is long type , hence size of union is 8 bytes .

The total memory required :10+8 =18 bytes .

3)

What does the given program print?

char c[ ] = “GATE2011”

char \*p = c;

printf (“%s”, p + p[3] – p[1]);

GATE CSE 2011

1. GATE 2011

2. 2011

3. E2011

4. 011

SOLUTION :

The correct option is 2 .

Given string is “GATE2011”

P=G

P[3]=E

P[1]=A

🡪P+P[3]-P[1]=P +‘E’-‘A’

[ASCII values of E= 69 , A=65]

🡪P+69-65=P+4=move 4 characters ahead

Hence,2011 is the answer .

4)

The output of the following C program is\_\_\_\_\_\_\_\_\_\_

GATE CSE 2015 Set 1

void f1(int a, int b) {

int c;

c=a; a=b; b=c;

}

void f2(int \*a, int \*b) {

int c;

c=\*a; \*a=\*b; \*b=c;

}

int main(){

int a=4, b=5, c=6;

f1(a,b);

f2(&b, &c);

printf(“%d”,c-a-b);

}

SOLUTION :

The output for the above program is -5 .

5)

The following program prints \_\_\_\_\_\_\_\_\_\_\_

#include <stdio.h>

void f (int \*p, int \*q) {

p = q;

\*p = 2;

}

inti = 0, j = 1;

int main ( ){

f(&i, &j);

printf (“%d %d \ n”, i, j);

return 0;

}

GATE CSE 2010

1. 2 2

2. 2 1

3. 0 1

4. 0 2

SOLUTION :

The correct option is 4 .

The output for the above program is 0 2 .

6)

The output of the following C program is

GATE CSE 2018

#include<stdio.h>

structOurnode{

char x,y,z;

};

int main(){

structOurnode p = {‘1’, ‘0’, ‘a’+2};

structOurnode \*q = &p;

printf (“%c, %c”, \*((char\*)q+1), \*((char\*)q+2));

return 0;

}

1. 0, c

2. 0, a+2

3. ‘0’, ‘a+2’

4. ‘0’, ‘c’

SOLUTION :

The correct option is 3 .

The output for the above program is ‘0’ , ‘a+2’ .

7)

The output of the following C program is

#include <stdio.h>

void mystery(int \*ptra, int \*ptrb) {

int \*temp;

temp = ptrb;

ptrb = ptra;

ptra = temp;

}

int main() {

int a=2016, b=0, c=4, d=42;

mystery(&a, &b);

if (a < c)

mystery(&c, &a);

mystery(&a, &d);

printf(“%d\n”, a);

}

SOLUTION :

The output for the above program is 2016.

8)

The output of the following C program is

#include <stdio.h>

int main () {

intarr [] = {1,2,3,4,5,6,7,8,9,0,1,2,5}, \*ip = arr+4;

printf (“%d\n”, ip[1]);

return 0;

}

SOLUTION :

The output for the above program is 6 .

9)

Consider the following C function

GATE CSE 2004

void swap (int a, int b)

{

int temp;

temp = a;

a = b;

b = temp;

}

In order to exchange the values of two variables x and y

1. call swap (x,y)

2. call swap (&x, &y)

3. swap (x, y) cannot be used as it does return any value

4. swap (x,y) cannot be used as the parameters are passed by value

SOLUTION :

The correct option is 2 .

call swap(&x,&y) is used to swap the values of x and y as it can modify the values with respect to addresses .