**Hassaan Akbar Cheema**

**174351**

**Task1**

1. Why does the memory address stored in pointer “pa” vary by 4?

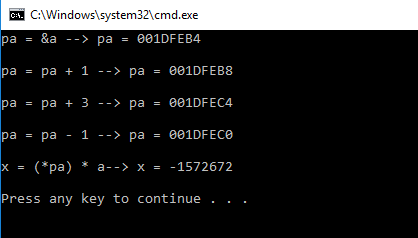
Ans: pa is a pointer that is pointing to the location of a. pa is an integer type pointer and integer takes 4 bytes of memory by default. If we add 1 to the pointer address it will move to the very next value in the memory which is stored 4 bytes below the first. That is why the integer type pointer vary by 4 when incremented by 1.

1. Will the address still vary by 4 if the data type of the above mentioned code changed from “int” to “long”? Explain your answer.

Ans: yes , if the data type is changed from int to long it will still vary by 4 because long also takes 4 bytes of memory.

1. If we try to multiply the address pointed to by “pa” what will happen? Is this logically or programmatically correct? Attach screen shot of the output you get when you try this multiplication.

Ans:



**Task2**

// ConsoleApplication2.cpp : Defines the entry point for the console application.

//

#include "stdafx.h"

#include <iostream>

using namespace std;

int sum(int a[], int size){

int res = 0;

int \*p;

for (int i = 0; i < size; i++){

p = &a[i];

res += \*p;

p++;

}

return res;

}

int check(int \*a, int \*b, int size){

int \*p, \*c, \*d;

int res[10];

c = a;

d = b;

for (int i = 0; i < size; i++)

{

p = &res[i];

c = &a[i];

d = &b[i];

\*p = \*c + \*d;

}

return sum(res, size);

}

int \_tmain(int argc, \_TCHAR\* argv[])

{

const int s = 10;

int res;

int \*p = new int[s];

int \*q = new int[s];

for (int i = 0, j = 1; i < s; i++, j++){

p[i] = i;

}

for (int i = 0, j=10; i < s; i++, j--){

q[i] = j;

}

res = check(&p[0], &q[0], s);

cout << "the result is " << res << endl;

return 0;

}

