**Data Structures and Algorithms**

**Name:** Ahmad Amjad Mughal

**Reg No:** 121672

**Class:** BSCS-6C

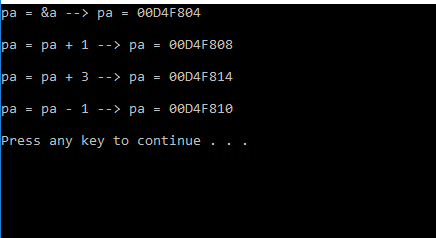
**Task1:**

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

**Answer:** Yes, pointed address varies by 4 because int datatype variable have a size of (1 byte = 4 bits). if we do increment, decrement or any addition or subtraction resulted pointed address will vary by byte.

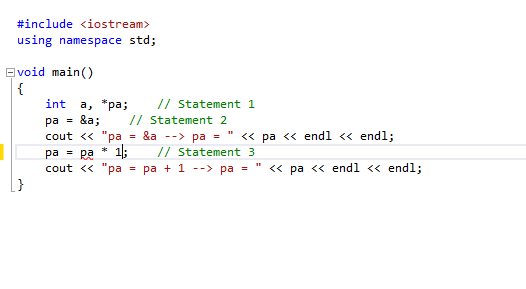
**Question 2:** Will the address still vary by 4 if the data type of the above mentioned code changed from “int” to “long”? Explain your answer.

**Answer:** Yes, the address still varies by 4 if we change the data type for the code. The reason is that pointers store the address of variable that it points. Space occupies by **int** and **long** variable that is to be pointed by any pointer its address will change in a same order.in case of increment in variable its pointed address will vary by 4.



**Question 3:** 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.

**Answer:** You can't perform multiplication and division on pointers. If we multiply the pointed address by any nonzero number, then program will not run as it causes errors. It is programmatically incorrect.



There occur few programming errors



**Task 2:**

**Code:**

//Ahmad Amjad mughal

//Regg No:121672

//Class:BSCS-6C

//Task2

#include<iostream> //including library

using namespace std;

int sum(int \*a, int size) //function sum that uses pointer and size as a variable

{

int res = 0;

for (int i = 0; i<size; i++) //loop that continues till i = 10

res += a[i];

return res; //it returns result to the calling function

}

int check(int \*a, int \*b, int size){ //another function with parameters pointers \*a and \*b and size as an interger variable

int \*res; //definition of pointer

res = new int[size]; //the use of New clause this is dynamic memory allocation

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

{

res[i] = a[i] + b[i]; //performing addition

}

return sum(res, size); //returns a function call

}

int main(void){ //main function

const int s = 10;

int res;

int array1[s] = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };

int array2[s] = { 10, 9, 8, 7, 6, 5, 4, 3, 2, 1 };

res = check(array1, array2, s);

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

return 0;

}