# Question 4: Array

**Q4.1: Write program to ask the user to type 10 integers of an array. The program must compute and write the number of integers greater or equal to 10.**

Code:

#include<iostream>

using namespace std;

#define n 10

int count(int a[]) {

int dem = 0;

for (register int i = 0; i < n; i++) {

if (a[i] >= 10) dem++;

}

return dem;

}

int main(int argc , char\*\* argv) {

int array[n];

for (register int i = 0; i < n; i++) {

cout << "Nhap so thu " << i + 1 << " : ";

cin >> array[i];

}

cout << "So phan tu >= 10 la : " << count(array) << endl;

system("pause");

return 0;

}

**Q4.2: Write a program which takes 2 arrays of 10 integers each, a and b. c is an array with 20 integers. The program should put into c the appending of b to a, the first 10 integers of c from array a, the latter 10 from b. Then the program should display c.**

* **Use memcpy**
* **Without memcpy**

Code:

#include<iostream>

using namespace std;

void use\_memcpy(int a[], int b[], int c[]);

void without\_memcpy(int a[], int b[], int c[]);

void show(int a[], int n);

int main(int argc, char \*\* argv) {

int a[10] = { 0,2,4,5,3,6,5,8,4,9 };

int b[10] = { 5,6,4,7,8,2,26,5,4,8 };

int c[20] = { 0 };

use\_memcpy(a, b, c);

//without\_memcpy(a, b, c);

show(a, 10);

show(b, 10);

show(c, 20);

system("pause");

return 0;

}

void use\_memcpy(int a[], int b[], int c[]) {

memcpy(c, a, sizeof(int) \* 10);

memcpy(c + 10, b, sizeof(int) \* 10);

}

void without\_memcpy(int a[], int b[], int c[])

{

for (register int i = 0; i < 10; i++) {

c[i] = a[i];

c[10 + i] = b[i];

}

}

void show(int a[], int n) {

for (register int i = 0; i < n; i++) {

cout << a[i] << " ";

}

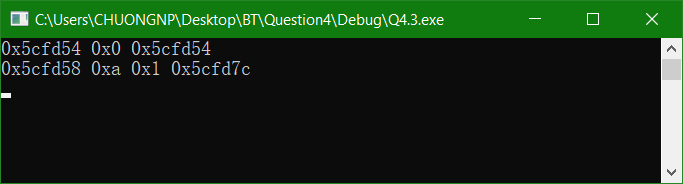
cout << endl;

}

**Q4.3: Explain the result:**

|  |
| --- |
| #include <stdio.h>  #include <iostream>  int main(int n, char\*\* args)  {  int a[10];  for (int i = 0; i < 10; i++)  {  a[i] = i\*10;  }  printf("0x%x 0x%x 0x%x\n", a, \*a, &a);  printf("0x%x 0x%x 0x%x 0x%x\n", (a + 1), \*(a + 1), \*a + 1, &a + 1);  } |

Kq: (có thể khác nhau sau mỗi lần chạy)



* Dòng đầu tiên: printf("0x%x 0x%x 0x%x\n", a, \*a, &a);

+) Giá trị đầu là địa chỉ phần tử đầu tiên của mảng a ở hệ thập lục phân.

+) Giá trị thứ hai là giá trị của a[0] ở hệ thập lục phân (a[0] = 0 vì sau thi thực hiện câu lệnh a[i] = i\*10; trong vòng lặp for).

+) Giá trị thứ 3 là địa chỉ của mảng a (&a lấy địa chỉ mảng a).

* Dòng 2: printf("0x%x 0x%x 0x%x 0x%x\n", (a + 1), \*(a + 1), \*a + 1, &a + 1);

+) Giá trị đầu là địa chỉ của phần tử thứ hai của mảng a. ( vì a là mảng kiểu int nên khi dịch 1 phần tử tức là dịch đi 4 bytes)

+) Giá trị thứ hai là giá trị mà phần tử thứ hai đang lưu giữ ở hệ thập lục phân (a[1] = 10 (= 0xa) vì sau thi thực hiện câu lệnh a[i] = i\*10; trong vòng lặp for).

+) Giá trị thứ 3: \*a + 1 tức là lấy giá trị phần tử đầu tiên + 1 => kq là 0x1 (phần tử đầu tiên của a = 0).

+) Giá trị cuối cùng là địa chỉ byte liền sau byte cuối cùng của mảng a (&a + 1 dịch một số byte bằng với kích thước của mảng).

**Q4.4: Implement bubble sort algorithm (sort acceding) to sort an array from “offset” with “count” elements.**

**For example: a[5] = {100, 4, 2, 6, 3} 🡪 sort(a, 1, 4) 🡪 {100, 2, 3, 4, 6}**

**Code:**

#include<iostream>

using namespace std;

#define max 100

void BubbleSort(int arr[], int offset, int count);

void Show(int a[], int n);

int main(int argc, char \*\*argv) {

int a[max];

int n, offset;

cout << "nhap so luong phan tu : n = ";

cin >> n;

for (register int i = 0; i < n; i++) {

cout << "nhap phan tu thu " << i + 1 << " : ";

cin >> a[i];

}

do {

cout << "nhap vi tri bat dau sap xep (offset<n): offset = ";

cin >> offset;

} while (offset > n);

Show(a, n);

BubbleSort(a, offset, n);

Show(a, n);

system("pause");

return 0;

}

void BubbleSort(int arr[], int offset, int count)

{

int temp;

for (register int i = offset; i < count - 1; i++) {

for (register int j = i + 1; j < count; j++) {

if (arr[j] < arr[j-1]) {

temp = arr[j];

arr[j] = arr[j-1];

arr[j-1] = temp;

}

}

}

}

void Show(int a[], int n)

{

for (register int i = 0; i < n; i++) {

cout << a[i] << " ";

}

cout << endl;

}

**Q4.5: Implement functions to invert member of an array**

* **void Invert(int input[], int num\_element): invert input array.**
* **void Invert(int input[], int num\_element, int output[]): generate output array from inverted input. Note: after executing this function, input array is not changed**

**Code:**

#include<iostream>

using namespace std;

void Invert(int input[], int num\_element);

void Invert(int input[], int num\_element,int output[]);

void Show(int arr[],int n);

int main(int argc, char \*\*argv) {

int a[10] = { 0,2,4,5,3,6,5,8,4,9 };

int b[10];

Show(a,10);

Invert(a, 10);

Show(a,10);

Invert(a, 10, b);

Show(b, 10);

system("pause");

return 0;

}

void Invert(int input[], int num\_element)

{

int n = num\_element / 2,temp;

for (register int i = 0; i <= n; i++) {

temp = input[i];

input[i] = input[num\_element - i - 1];

input[num\_element - i - 1] = temp;

}

}

void Invert(int input[], int num\_element, int output[])

{

for (register int i = 0; i < num\_element; i++) {

output[num\_element - 1 - i] = input[i];

}

}

void Show(int arr[], int n)

{

for (register int i = 0; i < n; i++) {

cout << arr[i] << " ";

}

cout << endl;

}

**Q4.6: Explain the result:**

|  |
| --- |
| #include <iostream>  using namespace std;  int x = 11;  int y = 12;  int a[5] = {21, 22, 23, 24, 25};  int u = 31;  int v = 32;  void main()  {  for (int i = -2; i < 7; i++)  cout << i << " " << a[i] << endl;  } |

**Kq:**

