# **CE100: Computing Fundamentals & Programming**

**Assignment # 9**

**Name: Abubakar Saif Roll Number: BSCE21017**

### **Instructions:**

• Email instructor or TA in case you are facing any difficulty in writing algorithm of any question.

• You cannot look at others’ solution or use others’ solution, however, you can discuss it with each other.

• Plagiarism will lead to a straight zero in all previous assignments with additional consequences as well.

• Code should be yours, not from your internet or any other source.

If any instance is found copied from an online source, it will be also considered as cheating.

• Submission after due time will not be accepted.

• From no onward, heavy penalty will be there for un-indented code and codes with improper comments.

• Display appropriate input and output messages whether mentioned in question or not.

• Dry run your solution. It helps in finding out bugs in your program.

## **REPORT**

Solve the following and comment on every step of your algorithm as if you are explaining your code to your fellows to demonstrate clear understanding. Submit the PDF of the report containing code and screenshot of output.

### **TASK 1**

Write a function that searches for a number in an array and returns the number of times that number appears in the given array.

Array:

11 7 2 11 11 3

Search: 11

Output: 3 // It shows 11 appears 3 times in given array

Code:

**int** searchArray(**int** arr[TASK], **int** num) { *//searches the desired number in 1D array*

**int** count = 0;*//sets count to ZERO*

**for** (**int** i = 0; i < TASK; i++) { *//array control loop*

**if** (arr[i] == num) { *//if any element of array is equal to "number to be searched"*

count++; *//then increment 1 in count*

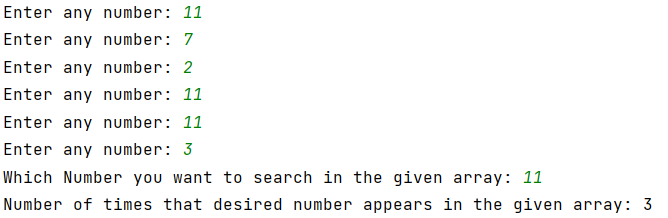
}

}

**return** count; *//passes the number of times that desired number appears in the array*

}

Output:



### **TASK 2**

Twenty-five numbers are entered from the keyboard into an array. Write a function to find out how many of them are positive, how many are negative, how many are even and how many are odd.

Array: (of size 7 for understanding)

2 -4 3 10 7 -1 -13

Code:

**int** countEven(**int** arr[TASK]) { *//count number of even inputs*

**int** count = 0;*//sets count to ZERO*

**for** (**int** i = 0; i < TASK; ++i) { *//array control loop*

**if** (arr[i] % 2 == 0) { *//if the remainder is ZERO when element of array is divided by 2*

count++;*//then add 1 in count*

}

}

**return** count;*//passes number of even inputs*

}

**int** countOdd(**int** arr[TASK]) { *//count number of odd inputs*

**int** count = 0;*//sets count to ZERO*

**for** (**int** i = 0; i < TASK; ++i) {*//array control loop*

**if** (arr[i] % 2 != 0) {*//if the remainder is not ZERO when element of array is divided by 2*

count++;*//then add 1 in count*

}

}

**return** count;*//passes number of even inputs*

}

**int** countPos(**int** arr[TASK]) {*//count number of positive inputs*

**int** count = 0;*//sets count to ZERO*

**for** (**int** i = 0; i < TASK; ++i) {*//array control loop*

**if** (arr[i] > 0) {*//if the element of array is larger than ZERO*

count++;*//then increment 1 in count*

}

}

**return** count;*//passes number of positive inputs*

}

**int** countNeg(**int** arr[TASK]) {*//count number of negative inputs*

**int** count = 0;*//sets count to ZERO*

**for** (**int** i = 0; i < TASK; ++i) {*//array control loop*

**if** (arr[i] < 0) {*//if the element of array is smaller than ZERO*

count++;*//then increment 1 in count*

}

}

**return** count;*//passes number of negative inputs*

}

**void** countAll(**int** arr[TASK]) {*//hub of all count functions*

**int** odd = 0, even = 0, pos = 0, neg = 0;*//declares variables needed*

odd = countOdd(arr); *//saves the value obtained from function in the variable*

even = countEven(arr);*//saves the value obtained from function in the variable*

pos = countPos(arr); *//saves the value obtained from function in the variable*

neg = countNeg(arr); *//saves the value obtained from function in the variable*

cout << **"Odds: "** << odd << endl; *//prints value stored in the variable*

cout << **"Evens: "** << even << endl; *//prints value stored in the variable*

cout << **"Positive: "** << pos << endl; *//prints value stored in the variable*

cout << **"Negative: "** << neg << endl; *//prints value stored in the variable*

}

Example Output:

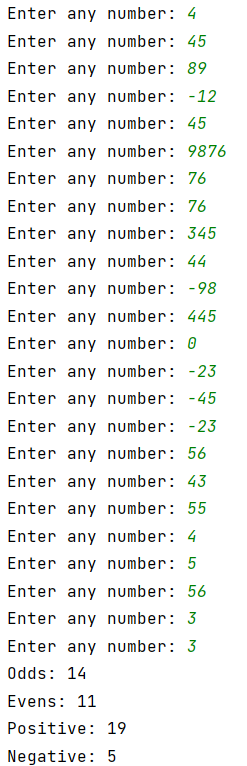
Odds: 4

Evens: 3

Positive: 4

Negative: 3

Screenshot:



### **TASK 3**

Create a function that finds and displays the smallest and largest number from the given 2D array.

Code:

**void** small2dArray(**int** arr[ROWS][COLS]) {

**int** small = arr[0][0], large;

**for** (**int** i = 0; i < ROWS; i++) {

**for** (**int** j = 0; j < COLS; j++) {

**for** (**int** i = 0; i < COLS; i++) {

**if** (small > arr[i][j]) {

small = arr[i][j];

}

}

large = small;

**for** (**int** i = 0; i < COLS; i++) {

**if** (large < arr[i][j]) {

large = arr[i][j];

}

}

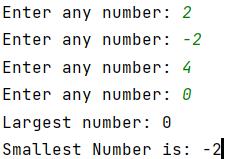
}

}

cout<<**"Largest number: "**<<large<<endl<<**"Smallest Number is: "**<<small<<endl;

}

Output:



### **TASK 4**

Create a program that checks each element of array and displays whether it’s prime or not.

Code:

**void** checkPrime(**int** arr[NUM]) { *//checks whether number is prime or not*

**int** prime = 0, count = 0; *//declares variable*

**for** (**int** i = 0; i < NUM; i++) { *//array control statement*

**if** (arr[i] == 0 || arr[i] == 1) { *//condition for if arr[i] is equal to zero or one*

cout << arr[i] << **" is not a Prime number"** << endl; *//tells that it is prime number*

} **else** { *//condition for if arr[i] is not equal to zero or one*

**for** (**int** j = 2; j <= arr[i]; j++) { *//executes loop until J will not be equal to number itself*

**if** (arr[i] % j == 0) { *//if remainder,when array element is divided ny j, is ZERO*

prime++;*//then prime will be incremented by 1*

}

}

**if** (prime == 1 && arr[i] != 1 && arr[i] != 0) { *//if prime is equal to ONE*

cout << arr[i] << **" is a Prime number"** << endl; *//then program will tell that it is prime number*

} **else** cout << arr[i] << **" is not a Prime number"** << endl; *//else prints it is not prime*

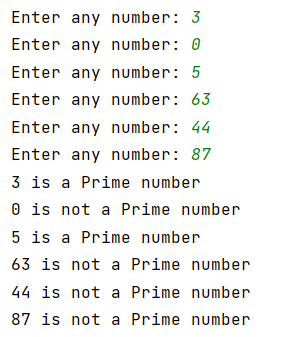
}

prime = 0; *//sets prime to ZERO for proper functioning of this code*

}

}

Output:



### **TASK 5**

Write a program that Reverse Array using pointers.

Code:

**void** reversePntArray(**int** arr[NUM]) { *//reverses the given array using pointers and DMA*

**int** \*pnt, j = NUM - 1; *//declares pointer and variable which is equal to array control variable - 1*

**int** \*reverse = **new int**[NUM]; *//declares DMA*

pnt = arr; *//directs pointer to arr*

cout << **"Original: "** << endl;

**for** (**int** i = 0; i < NUM; i++) {

cout << arr[i]; *//prints original array*

**if** (i != NUM - 1) { *//commas controlling loop*

cout << **","**; *//prints array*

}

\*pnt = arr[i]; *//stores arr[i] in the location pointed by the array*

reverse[j] = \*pnt; *//now store the value in pointed location in J index of DMA*

j--; *//decrement j by 1 in order to store numbers in reverse order*

}

cout << endl << **"Reverse: "** << endl;

pnt = arr; *//directs pointer to arr*

**for** (**int** i = 0; i < NUM; i++) {

pnt = arr + i; *//directs pointer to arr + i*

\*pnt = reverse[i]; *//stores reverse[i] in the location pointed by the array*

cout << arr[i]; *//now print arr[i]*

**if** (i != NUM - 1) { *//commas controlling loop*

cout << **","**; *//prints array*

}

}

cout << endl;

**delete**[] reverse; *//deletes the memory allocated by DMA reverse*

}

Sample output:

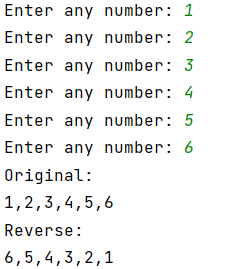
Original:

1, 2, 3, 4, 5

Reverse:

5, 4, 3, 2, 1

Output Screenshot:



### **TASK 6**

Write a program to Sort Array using pointers.

Sample output:

Original:

21, 2, 13, 40, 5

Reverse:

2, 5, 13, 21, 40

Code:

**void** sortPntArray(**int** arr[NUM]) { *//sorts given array by pointer*

**int** small, large, \*pnt; *//declares pointer and variables*

**int** \*tempSort = **new int**[NUM]; *//declares DMA*

cout << **"Original: "** << **"["**;

**for** (**int** i = 0; i < NUM; i++) {

cout << arr[i]; *//print original array*

**if** (i != NUM - 1) { *//commas controlling loop*

cout << **","**;*//prints array*

}

}

cout << **"]"** << endl;

pnt = tempSort;*//directs pointer to tempSort*

**for** (**int** j = 0; j < NUM; j++) {

small = arr[0]; *//sets SMALL to arr[0]*

**for** (**int** i = 0; i < NUM; i++) {

**if** (small > arr[i]) { *//checks which number is smallest in the given array*

small = arr[i]; *//stores smallest number in SMALL variable*

}

}

large = small; *//sets large equal to small*

**for** (**int** i = 0; i < NUM; i++) {

**if** (large < arr[i]) { *//checks which number is largest in the given array*

large = arr[i]; *//stores largest number in LARGE variable*

}

}

**for** (**int** i = 0; i < NUM; i++) {

**if** (arr[i] == small) {

arr[i] = large;

}

}

**if** (tempSort[j-1] == small) {

pnt = (tempSort + j) - 1;

\*pnt = tempSort[j-2];

pnt = tempSort + j;

\*pnt = small;

}**else** {

pnt = tempSort + j;

\*pnt = small;

}

}

pnt = arr;

cout << **"After Sorting: "** << **"["**;

**for** (**int** i = 0; i < NUM; i++) {

pnt = arr + i;

\*pnt = tempSort[i];

cout << arr[i];

**if** (i != NUM - 1) {

cout << **","**;

}

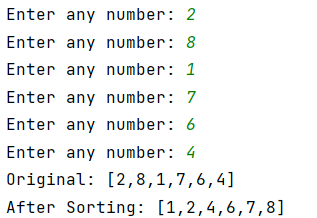
}

cout << **"]"** << endl;

**delete**[] tempSort;

}

Output:



### **Function prototypes**

Function prototype(s) should be written in Functions.h

### **Additional Functions**

Additional function(s) should be written in Functions.cpp and used in main.cpp

### **Run command**

make run This will run main.cpp executable

### **Notes**

* g++ can be used to compile and link C++ applications for use with existing test harnesses or other C++ testing frameworks.
* You should use C++ standard approach for the development, using g++ extensions is not acceptable

# **CE100: Computing Fundamentals & Programming**

### **Instructions:**

• Email instructor or TA in case you are facing any difficulty in writing algorithm of any question.

• You cannot look at others’ solution or use others’ solution, however, you can discuss it with each other.

• Plagiarism will lead to a straight zero in all previous assignments with additional consequences as well.

• Code should be yours, not from your internet or any other source.

If any instance is found copied from an online source, it will be also considered as cheating.

• Submission after due time will not be accepted.

• From no onward, heavy penalty will be there for un-indented code and codes with improper comments.

• Display appropriate input and output messages whether mentioned in question or not.

• Dry run your solution. It helps in finding out bugs in your program.

## **REPORT**

Solve the following and comment on every step of your algorithm as if you are explaining your code to your fellows to demonstrate clear understanding. Submit the PDF of report containing code and screenshot of output.

### **TASK 1**

Write a function that searches for a number in an array and returns the number of times that number appears in the given array.

Array:

11 7 2 11 11 3

Search: 11

Output: 3 // It shows 11 appears 3 times in given array

### **TASK 2**

Twenty-five numbers are entered from the keyboard into an array. Write a function to find out how many of them are positive, how many are negative, how many are even and how many are odd.

Array: (of size 7 for understanding)

2 -4 3 10 7 -1 -13

Output:

Odds: 4

Evens: 3

Positive: 4

Negative: 3

### **TASK 3**

Create a function that finds and displays the smallest and largest number from the given 2D array.

### **TASK 4**

Create a program that checks each element of array and displays whether it’s prime or not.

### **TASK 5**

Write a program that Reverse Array using pointers.

Sample output:

Original:

1, 2, 3, 4, 5

Reverse:

5, 4, 3, 2, 1

### **TASK 6**

Write a program to Sort Array using pointers.

Sample output:

Original:

21, 2, 13, 40, 5

Reverse:

2, 5, 13, 21, 40

### **Function prototypes**

Function prototype(s) should be written in Functions.h

### **Additional Functions**

Additional function(s) should be written in Functions.cpp and used in main.cpp

### **Run command**

make run This will run main.cpp executable

### **Notes**

* g++ can be used to compile and link C++ applications for use with existing test harnesses or other C++ testing frameworks.
* You should use C++ standard approach for the development, using g++ extensions is not acceptable