


National University of Computer and Emerging Sciences, Lahore Campus

	Course Name:	Programming Fundamentals	Course Code:	CS
	Degree Program:	CS	Semester:	Fall 2019
	Exam Duration:	180 Minutes	Total Marks:	50
	Paper Date:	12-Dec-19	Weight	
	Section:	ALL	Page(s):	5
	Exam Type:	Final Exam		

Student : Name: Muhammad Ahmad **Roll No.** 21L-5617 **Section:** A1

- Instruction/Notes:**
- Solve the exam on this question paper. Rough sheets can be used but will not be attached or marked.
 - In the programming problems, you are free to use the standard C++ library functions from iostream, cstring etc. where required.

Q1. [5 + 5] (a) Print the output of the following code:

<pre>void processNums(int nums[], int n) { int i=-1, j=0; for(;j<n; ++j){ if(nums[j]==1){ swap(nums[i+1],nums[j]); ++i; } } }</pre>	<pre>int main() { int nums[]={0, 1, 0, 0, 1, 1, 1, 0, 1, 0}; int n = 10; cout << "Original array: "; for (int i=0; i < n; i++) cout << nums[i] << " "; processNums(nums, n); cout << "\nProcessed array: "; for (int i=0; i < n; i++) cout << nums[i] << " "; return 0; }</pre>
<p>Output:</p> <p style="color: blue;">original array ;0100111010 processed array; 1111100000</p>	

(b) Print the output of the following code:

<pre>void UpdateStr(char str[]) { for (int i = 1; str[i] != '\0'; i++){ if (str[i] >= 'a' && str[i] <= 'z' && str[i-1] == ' ') str[i] = str[i] - 32; } }</pre>
<pre>int main() { char str[] = "I am a student of programming!"; UpdateStr(str); cout << str << endl; }</pre>
<p>Output:</p> <p style="color: blue;">I am a student of programing !</p>

Q2. [10 + 4]

(a) A black and white image is stored in a matrix *data* of dimensions $m \times n$. That is, it has m rows, n columns and $n \leq 1000$. Matrix *data* contains only 0s and 1s. 0 represents a black pixel and 1 represents white pixel. We are interested in counting the number of $k \times k$ blocks (sub-matrices) in *data* that have at least $k^2/2$ white pixels in them. We call such blocks “*mostly white blocks*”. Write a C++ function that returns the count of “*mostly white blocks*” in *data*.

You may use the function header: `int countMostlyWhite(int data[][1000], int m, int n, int k)`

For further clarity consider the following example: Here matrix *data* has dimensions $m = 5, n = 4$, and let's take $k = 3$. Take each 3×3 sub-matrix (this example has a total of 6 such matrices), and count its white pixels. Each “*mostly white block*” must have at least $3^2/2 = 4.5 \approx 5$ white pixels for $k = 3$. In the example there are 4 “*mostly white blocks*” of size 3×3 with their top-left corners at (0, 1), (1, 1) (2,0) and (2, 1) respectively.

0	0	1	0
0	1	0	1
0	0	0	1
1	0	1	0
0	1	1	1

```
#include<iostream>
using namespace std;
countMostlyWhite( int rows, int col )
{
    int data[100][100];
    cout << "0 represents the black pixel"<<endl;
    cout << "1 represents the white pixel"<<endl;
    data[rows][col];

    int count = 0 , total = 0 , result = 0,k;
    for(int i = 0 ; i <=rows ; i++)
    {
        for(int i = 0 ; i <= col ; i++)
        {
            cout<<"Enter the values for pixels (0 and 1) : ";
            cin >>data[rows][col];
        }
    }
    cout<<"Enter the value for k: ";
    cin>>k;
    total = k*k;
    result = total/2;
    cout<<"total mostly white blocks: ";
    cout << result<<endl;
    for(int i = 0 ; i <= k ; i++)
    {
        for(int j = 0 ; j <= k ; j++)
        {
            if(data[i][j] == 0)
            {
                cout<<"the co-ordinates where the mostly white block exist = ";
                cout << "("<<i<<" , "<<j<<")";
            }
        }
    }
    return 0;
}
int main()
{
    int r , c ;
    cout << "Enter the number of rows for the matrix = ";
    cin >> r;
    cout << "Enter the number of columns for the matrix = ";
    cin >> c;
    countMostlyWhite( r,c);
    return 0;
}
```

(b) Suppose in part (a) above we also asked you to report the top left coordinates (row number and col number of the top left cell) of each and every *mostly white blocks* in *data*. Write down the changed function header that would be required to accomplish this task. Particularly, what new parameters may be required? Explain the purpose of these parameters n two lines.

New parameters ; count ,total and result
 There are used to calculate the number of blokes white spaces and their coordinates

Q3. [6] Short Questions

(a) Write a cout statement that prints **H"e"llo** on the screen. char X = 34;

```
cout<<"H"<<X<<"e"<<X<<"llo"<<endl;
```

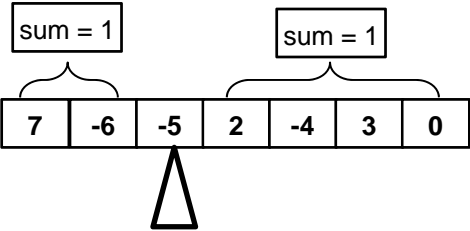
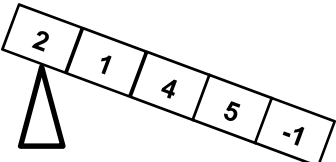
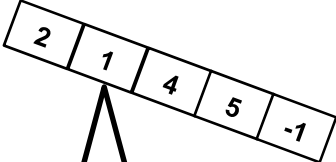
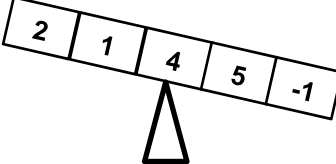
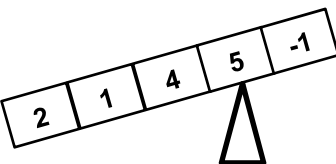
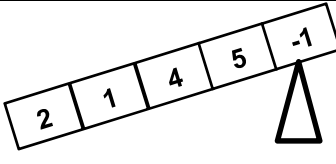
(b) Properly indent the following code.
 int a=1,b=2;if(a<b)cout<<"first"; else cout<<"second";

```
int a =1,b = 2;
if(a<b)
    cout<<"first";
else
    cout<<"second";
```

(c) What is the binary equivalent for (1010)₁₀ (number given in decimal system)?

```
in binary it is  1111110010
```

Q4. [10] Write a C++ function called **findEquilibrium** that accepts an integer array A of size n and finds the *equilibrium index* of the array if it exists. An index k of an array is said to be an equilibrium index if the sum of all the elements on the left of k is equal to the sum of all the elements on the right of k. Precisely, k is an equilibrium index of A if and only if $\sum_{i=0}^{k-1} A[i] = \sum_{i=k+1}^{n-1} A[i]$. For clarity, consider this example below. If the index is not found, the function returns -1.

Equilibrium Index Found	Equilibrium Index not Found
	 <p>[0] is not equilibrium index</p>  <p>[1] is not equilibrium index</p>  <p>[2] is not equilibrium index</p>  <p>[3] is not equilibrium index</p>  <p>[4] is not equilibrium index</p>
findEquilibrium returns 2 since equilibrium found at index number 2.	findEquilibrium returns -1 since equilibrium not found at any index 0, 1, 2, 3 and 4.

```

#include <iostream>
using namespace std;
int Find_Equilibrium(int Arry[],int Length)
{
    int sum1,sum2;
    for(int i=0;i<Length;i++)
    {
        sum1=0;
        for(int j=0;j<i;++j)
            sum1=sum1+Arry[j];
        sum2=0;
        for(int k=i+1;k<Length;k++)
            sum2=sum2+Arry[k];
        if(sum1==sum2)
        { cout<<"the given array is stable at given index = ";
          return i;
        }
    }
    cout<<"its not a stable array";
    return -1;
}
int main()
{
    int a[100],size;
    cout<<"enter the size of array = ";
    cin>>size;
    for(int i =0;i<size;i++){
        cout<<"enter "<<i<<" element = ";
        cin>>a[i];}
    cout<<Find_Equilibrium(a,size);

    return 0;
}

```

Q5. [10] Morse code is an encoding scheme used in telecommunication to transmit text messages from one point to the other. The encoding scheme simply uses combinations of dots and dashes to represent alphabetic characters. We consider a fixed length code: each letter is represented by 5 symbols, as shown in the code table below.

A	B	C	D	E	F	G	H	I	J	K	L	M
N	O	P	Q	R	S	T	U	V	W	X	Y	Z

The code for space is Write a C++ function that can takes in a message in the form of fixed-length Morse code and converts it to English text. You need to decide the input parameters of the function. Sample example:

Input message: -.....-..-...-.....-...-.....-..-..-.....-...-..-...-..-..

Output: THIS IS MESSAGE

Note: this code table is one of the parameters to your function (you do not need to initialize it yourself). It is a 2-D array of the type: **char table [27] [6]**, where row table[0] is a NULL-terminated character array of length 5 containing the code for A, table[1] containing code for B, and so on.

```
#include<iostream>
#include<string>
using namespace std;
int main()
{
    char Table[27][6];
    string n;
    string text;
    cout<<"Enter the morse code: ";
    cin>>n;
    for(int i = 0 ; i <= 26 ; i++)
    {
        if(i==1)
        {
            if(n == Table[i]);
            text = 65;
        }
        else if(n == Table[i])
            text = 65 + i;
    }
    cout<<"The Converted code is: "<<text;
    return 0;
}
```