CS 101: Introduction to Computing

Sessional II, Fall 2013

Date: Nov 1st 2013 Marks: 40 Time: 90 minutes

Name: ________________________ Roll No: ________________________ Section: _________________

INSTRUCTIONS:

- You may ask for extra sheet for rough work, however, it will ***not*** be collected
- Make sure your provide answers ***in*** the space provided. I follow this rule quite strictly!
- Follow the conventions discussed in class, correctness alone will not earn you full marks

Q1 - Consider the following C++ program. What is the output from the program in response to the following user input?

```
#include <iostream>
using namespace std;
int main()
{
       int n, m;
       cout << "Please enter two integers: ";
       cin >> n >> m;//first entered value gets stored in n, second in m
       if (n > m)
              cout << n % m << endl;
       else
       {
              for (int r = 1; r < n; r++)
              {
                     for (int c = 1; c < m - n - 1; c++)
                     {
                            cout << "*+";
                     }
                     cout << endl;
              }
       }
       return 0;
}
```

(i) The user enters: 10 9

(ii) The user enters: 3 7

(iii) The user enters: 3 15

(iv) The user enters: 10 15

(v) The user enters: -1 5

Q2 - Just Roll with it~

Remember the good old dice? It has 6 faces with numbers 1-6 on it. Implement a game that asks a user to input two scores that represent the numbers on two different dice. If any of the number is incorrect, then the game should output “invalid dice number” and ask for the score again.

So after you have your valid inputs, here are the rules: If the sum of the scores on the two dice is even or 3, the program outputs “you are a winner” and the game ends. If the sum of scores of the two dice is 11 then the program should output “you have lost the game” and the game ends. If the sum of scores of the two dice is 5, 7 or 9 then the program should output “carry on” and asks for the scores again (the game begins again).

Q3 - Strong coding Arm (and eye) required!

The following code should determine whether a number is an Armstrong number or not. An Armstrong number is a number which is equal to the cube of its individual digits. For example 371 = $3^3+7^3+1^3$

There are logical errors in the following code. Explain what they are and how you may correct them.

```
#include <iostream>
using namespace std;
int main()
{
       int x,quotient=0,remainder,sum,number;
       cout << "Input a number ";
       cin >> x;
       number = x;
       remainder = x;
       sum=0;
       while (x < 10)
       {
              quotient = x/10;
              remainder = x%10;

              x = quotient;
              sum = sum+ quotient*quotient*quotient ;
       }

       sum = sum+ remainder*remainder*remainder;

       if (sum==number)
              cout << "I have found an amazing Armstrong number";
       else
              cout << "This is not my amazing Armstrong number";

       return 0;
}
```

Q4 - Who is the *Popular* of 'em all?

You are given an integer array intArray and its size MAX_SIZE. Write a program which determines whether there is any number in intArray which occurs *more than* 50% of times (that is, *more than* ceiling(n/2) times) – such an element is called a Popular Element. So if n =10, and one of the elements in intArray occurs more than 5 times, then it is a Popular element. If n=11, and one of the elements occurs more than 6 times then it is a Popular element, for example:

1 1 1 4 5 1 7 8 1 1 (array size is 10 and 1 occurs more than 5 times. 1 is a popular element)
2 2 2 4 2 1 7 8 2 2 2 (array size is 11 and 2 occurs more than 6 times so 2 is a popular element)
The program should either print the popular element or report that none exists.

```
#include <iostream>
using namespace std;
int main()
{
        //Assume the code to declare and initialize the array is already given here
        //from this point you have a declared and populated integer array by the name intArray and
        //its size is given to you in an integer MAX_SIZE
```

(Question 4 continued)