EECE 230 Introduction to Programming, Sections 1, and 2
Quiz II

April 14, 2014

- The duration of this exam is 1 hour and 50 minutes.
- It consists of 4 problems.
- The exam is closed book. You can use also all the material on Moodle: lecture notes, programming assignments, and solutions, etc. You are **NO**T allowed to use the **web** (**imail** included), You are not allowed to use **USB's** or files previously stored in your **account**.
- If you violate the above rules or if you communicate with a person other than the exam proctors during the exam, you will immediately get zero and you will be referred to the appropriate disciplinary committee.
- Active cell phones and any other unauthorized electronic devices are absolutely not allowed in the exam rooms. They should be turned off and put away.
- Plan your time wisely. Do not spend too much time on any one problem. Read through all of them first and attack them in the order that allows you to make the most progress.
- Submit your solutions each part in a separate file as indicated in the booklet. Include your name and ID number in each file. Submit the files online in a single zip file called $yourLastName.yourFirstName.zip$.
- Good luck!

Problem 1 (30 points). Strings: Flip halves

Write the function $flipHalves$ which takes as input parameters two C-strings s and t. It is supposed to store in t a flipped version of s constructed as follows. If the length of s is even, then t should contain the second half of s followed by its first half. That is, if $s = uv$, where u and v are each of length $n/2$ and n is the length of s, then the function should set t to $t = vu$. If the length n of s is odd, say $s = ucv$, where u and v are each of length $(n-1)/2$ and c is a single character, then the function should set t to $t = vcu$. Check the examples below. Assume that enough memory is allocated to t before calling the function.

You can use the *strlen* function.

Use the following test program:

```
#include<iostream>
using namespace std;

//flipHalves prototype

int main()
{
      char s1[] = "abcdefgh", t1[10];
      flipHalves(s1,t1);
      cout<<"t1: "<<t1<<endl;

      char s2[] = "abcde", t2[10];
      flipHalves(s2,t2);
      cout<<"t2: "<<t2<<endl;

      char s3[] = "ab", t3[10];
      flipHalves(s3,t3);
      cout<<"t3: "<<t3<<endl;


      char s4[] = "a", t4[10];
      flipHalves(s4,t4);
      cout<<"t4: "<<t4<<endl;

      return 0;
}
// flipHalves  body
```

You are supposed to get

```
t1: efghabcd
t2: decab
t3: ba
t4: a
```

Any correct solution is worth full grade. Submit this problem in a file called Prob1.cpp. Include your name and ID number in the file.

Problem 2 (30 points). Find outcast

We are given an array of integers $A[0 \ldots n-1]$. We want to check whether A contains an integer x such that all the other integers in A are far from x by at least 10. That is, given A and n, check whether

there exists $0 \leq i \leq n-1$, such that $|A[i] - A[j]| \geq 10$, for all $0 \leq j \leq n-1$ such that $j \neq i$. If such i exists, return any such i. Otherwise, we return -1. See the examples below.

Write a function to solve this problem. The function prototype is:

```
int findOutcast(int A[], int n)
```

Use the following test program:

```
int A[] = {55,50, 12, 1,0,31,39,23,25,49,43}; // 12 is the outcast and its index is 2
cout<<findOutcast(A,11)<<endl;

int B[] = {1,10,2,13}; // there is no outcast
cout<<findOutcast(B,4)<<endl;

int C[] = {12,100,30};// they are all outcasts
cout<<findOutcast(C,3)<<endl;
```

You should get

```
2
-1
0
```

Any correct solution is worth full grade. Submit this problem in a file called Prob2.cpp. Include your name and ID number in the file.

Problem 3 (30 points). Repeated number

We are given a sorted array $A[0 \ldots n-1]$ of consecutive integers. We know that they are distinct except one which appears twice. We would like to find the repeated number. That is, $A[] = \{a, a+1, a+2, \ldots, k-2, k-1, k, k, k+1, k+2, \ldots, b-1, b\}$ for some integers $a \leq k \leq b$. Thus k is the repeated integer. See the examples below. Write a function $findRepeated$, which given A and its size n, returns the repeated number k.

Your function is not supposed to work properly if the A is not as above (i.e., consists of consecutive integer which are are distinct except one which appears twice). Thus, in particular, you can assume that the length of A is at least 2.

Use the following test program:

```
int A[] = {10,11,12,13,13,14,15};
cout<<findRepeated(A,...
cout<<endl;

int B[] = {10,11,11,12,13,14,15};
cout<<findRepeated(B,...
cout<<endl;

int C[] = {10,10,11,12,13,14,15};
cout<<findRepeated(C,...
cout<<endl;

int D[] = {10,11,12,13,14,15,15};
cout<<findRepeated(D,...
cout<<endl;

int E[] = {10,10};
cout<<findRepeated(E,...
cout<<endl;
```

You should get

```
13
11
10
15
10
```

Any correct solution is worth 15/30. To get full grade, write an efficient implementation. You can do better than linear. Feel free to you use recursion if you believe it helps.

Submit this problem in a file called Prob3.cpp. Include your name and ID number in the file.

Problem 4 (25 points) Recursion

Consider the program

```
#include <iostream>
using namespace std;

void  main ()
{
cout<<"Enter n:";
int n;
cin>>n;
//////////////////////////
for(int i=0;i<=n;i++) {
    for(int j=0;j<=n;j++)
         cout<<i+j<<" ";
    cout<<endl;
    }
//////////////////////////
}
```

For instance $n = 5$, it displays the table:

```
Enter n: 5
0 1 2 3 4 5
1 2 3 4 5 6
2 3 4 5 6 7
3 4 5 6 7 8
4 5 6 7 8 9
5 6 7 8 9 10
```

Write a recursive version of the above program, i.e., replace the above for loops with recursion. The use of for or while loops is strictly prohibited in this problem.

Submit this problem in a file called Prob4.cpp. Include your name and ID number in the file.