EECE 230 Introduction to Programming, Sections 1,2 and 11
Quiz I

Oct 10, 2012

- The duration of this exam is 3 hours.
- It consists of 4 problems.
- You can use also all the material on the C++ tutorial website and on Moodle: lecture notes, programming assignments, and solutions, etc. You are **NOT** 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 on moodle in a single zip file called $yourLastName.yourFirstName.zip$.
- Good luck!

Problem 1 (20 points). Cedar radar.

You are the top programmer in the national Lebanese army. You got knowledge that an enemy drone follows a circular trajectory with a radius of $1Km$ around its target. The cedar radar system, built at AUB, can send you two coordinate sets for the drone (x_1, y_1) and (x_2, y_2). Write a program that takes the two sets of coordinates from the user (simulating the radar) and checks whether the drone is possibly surveying a secret target at position $(x_0, y_0) = (1.2km, 2.2km)$.

Note that the formula of a circle centered at (x_0, y_0) with radius R is $(x - x_0)^2 + (y - y_0)^2 = R^2$.

The program should request (x_1, y_1) from the user and check if the drone is possibly surveying the target. If so, then the program should request (x_2, y_2) and check again. If both checks match a circle around the target, the program should print a message with a warning. If one of the checks does not match a circle then the program should exit silently.

Sample input/output:

```
Enter x1,y1: 2.2 2.2
Enter x2,y2: 1.2 1.2
ALARM - EVICT SECRET POSITION

Enter x1,y1: 1 4

Enter x1,y1: 2.2 2.2
Enter x2,y2: 1 4
```

Submit your solution in a file called Prob1.cpp including your name and ID number.

Problem 2 (25 points). Avoid Paparazis.

A rockstar wants to avoid paparazis. She wants to leave her house at random timings so that the paparazis can not predict her outings. She bought a program from a secret agency that suggests two timings at random.

The following code uses functions `srand` and `rand` from the `stdlib` library to obtain two timings $(h1 : m1)$ and $(h2 : m2)$.

Function `rand` returns a pseudo-random integer each time it is called. Function `srand` is called usually once to ensure that the sequence of numbers returned by `rand` differs. Passing a different seed to `srand` to every program run is advisable. Using `rand` and `srand` is good for testing purposes and avoids requiring the user to enter values manually.

The values $h1$ and $h2$ denote hour values from 0 to 23 inclusive. The valyes $m1$ and $m2$ denote minute values from 0 to 59 inclusive.

```
#include <iostream>//needed for cout, endl
using namespace std;
#include <stdlib>// needed for rand and srand
int main() {
  int seed=0;
  cout << "Pleaze enter a seed number to randomize time." << endl;
  cin >> seed;
  srand(seed);

  int h1 = rand() % 24;//hours
  int m1 = rand() % 60;//minutes

  int h2 = rand() % 24;
  //The following 'if' check makes sure almost half the time
  // h2 is equal to h1 for testing purposes.
```

```
// It should be taken out in the final code.
if (rand() % 2) { // if rand () is even
  h2 = h1;}
int m2 = rand() % 60;

// insert your code here

return 0;}
```

(a) **10/25 pts.** The rock star decided to always pick the earlier suggested time. Write code that checks whether $t_1 = (h1 : m1)$ is smaller than $t_2 = (h2 : m2)$ and then advises the rock star when to leave. Insert your code where indicated in the comment. Note that t_1 is smaller that t_2 if

- $h1 < h2$, or
- $h1 = h2$ and $m1 < m2$.

(b) **15/20 pts**. The rock star asked your to modify the program as follows.

- The program asks her to provide the number of desired suggestings that is less than MAXSUGGESTIONS and stores it in `int n;`.
- The program computes '`n`' suggested timings (h:m) using `rand` and `srand` as shown above, and stores the timings in two arrays `int h[MAXSUGGESTIONS];` and `int m[MAXSUGGESTIONS];`.
- The program computes the earliest (minimum) timing `h[i]:m[i]` between 0 and $n-1$ inclusive and prints it for the rockstar.

Submit your solution to Part (a) in a file called Prob2a.cpp, and to Part (b) in a file called Prob2b.cpp including your name and ID number in each file.

Problem 3 (30 points). Lotto ticket.

In this problem you will write a program that simulates lottery drawings. The code below is a good starting structure, you may use it and modify it, and you may also start from scratch on your own.

The lottery works via the user selecting n numbers between 1 and 52. For the sake of the exam, you can preset these numbers to constants. We call these numbers the ticket $t = \{t_0, t_1, \ldots, t_{n-1}\}$. The value of n should be obtained from the user. The program should alarm the user and exit with failure if the user enters a bad number.

The program will compute pseudo-random numbers for the lotto drawing. We call these numbers the drawing $d = \{d_0, d_1, \ldots, d_{n-1}\}$. The program will use `rand` and `srand` from the `stdlib` library to compute the drawing; e.g. the values between 1 and 52 of each of the n balls. The function `srand` should be called only once and should take a seed value s from the user to seed the pseudo-random number generator. The function `rand` should be called n times to compute the drawings. A ticket is considered wining if it has more than 3 balls matching the drawing. The prize is relative to the number of matching balls.

1. **17 pts.** The program should now count the number of matching balls in the ticket. First assume that there are no duplicates in t and no duplicates in d. The number of wining balls is $|t \cap d|$.

2. **13 pts.** Now we want to take duplicates into account. When two balls t_i and d_j, $0 \leq i, j < n$ match, we increment the number of wining balls by one and we stop considering d_j when matching the rest of the balls $[t_{i+1} \ldots t_{n-1}]$.

3. **10,000 USD.** A new games company heard about your program. They offered you a good sum of money to use your progarm for simulation purposes. They want to simulate a full season of m ... (to be continued).

```
#include <iostream>//needed for cin, cout, endl
using namespace std;
#include <stdlib>// needed for rand and srand
```

```
const int CAPACITY = 12;

int main() {
  int seed=0;
  int n = 0;

  cout << "Pleaze enter a seed number to randomize the lotto drawing:" << endl;
  cin >> seed;
  srand(seed);

  cout << "Pleaze enter the number of balls in a ticket:" << endl;
  // Write code here

  int ticket[CAPACITY];
  // Write code here to initialize
  //  all the entries in ticke[0 .. n-1] to '1'

  int drawing[CAPACITY];
  //Write code to simulate the drawing of n balls
  // use rand: int x = rand() % X; returns a number between 0 and X.

  // write the rest of your code here

  return 0;}
```

Submit your solutions in files called Prob3a.cpp, and Prob3b.cpp for Parts A and B, respectively. Include your name and ID number.

Problem 4 (20 points). Estimate π.

The ratio of the area of a square inscribing a circle is $\pi/4$. Consider a circle centered at $(2.0, 2.0)$ with a radius of 1 and consider its inscribing square.

(a) **8 pts.** Use `rand` and `srand` from the `stdlib` library, along with appropriate arithmetic operations, to generate a point (x, y) where x and y are of type `double` such that the point (x, y) is inside the inscribing square. Note that `rand` returns a number of type `int` between 0 and `RAND_MAX`.

(b) **12 pts.** Repeat the oepration of Part (a) $10,000$ times and count the number of points that happen to be inside the circle. Print the ratio and validate whether it is a good estimate of $\pi/4$.

Submit your solution in a file called Prob4.cpp including your name and ID number.