Project #01 (18%) (Data Structures 2017)

Topics:
classes, object-oriented design.

In this lab you have to solve 7 uva.onlinejudge.org problems in object-oriented style.

Requirements:

- Do not use global variables (variables defined outside of any function).
- Do not use C arrays (int a[10]), C-strings (char s[] = "Hello"). Use standard C++ containers instead.
- Each function (including main) or method must have at most 25 lines.

Problem 01 (2%)

Solve UVA problem 101 "The Block Problem":
Create class RobotWorld with
constructor:
  RobotWorld(int n): n – number of blocks
and methods:
  void moveOnto(int a, int b)
  void moveOver(int a, int b)
  void pileOnto(int a, int b)
  void pileOver(int a, int b)
You may add as many methods as you want.
Use vector<vector<int>> as inner data structure for class RobotWorld;

Problem 02 (2%)

Solve UVA problem 540 "Team Queue".
Create class TeamQueue with
constructor:
  TeamQueue: creates empty queue
and methods:
  void enqueue(int id)
  int dequeue()
The implementation of the TeamQueue should be efficient: both adding and deleting element should only take constant time (do not depend on the number of elements in queue).

Use standard class vector and standard class queue as inner data structures of this class.

Problem 03 (2%)

Solve UVA problem 478:
Create abstract class Figure with abstract method bool contains(int x, int y);
Create derived classes Rect, Circle, Triangle using class Figure as a base. Implement corresponding methods bool contains(int x, int y) for each of these classes.
Use vector of shared pointers to store pointers to all figures in this program.

Problem 04 (2%)

Solve UVA problem 10196 “Check the Check” in object-oriented style:
Create abstract class Figure with abstract class method bool attacks(int row, int col);
Create derived classes King, Queen, Bishop … using class Figure as a base. Implement corresponding methods bool attacks(int row, int col) in each of these classes;
Create class Board to store current configuration of a game. Use vector of shared pointers to store pointers to all figures in this program.

Problem 05 (2%)

Solve UVA problem 495 “Fibonacci Freeze”.
Create class BigInt (integer type with arbitrary precision) with:
  constructor BigInt(): creates BigInt equal to 0;
  constructor BigInt(const string& s): creates BigInt equal to a given value s;
Define operators:
  BigInt operator+(const BigInt& a, const BigInt& b);
  ostream operator<<(ostream& out, const BigInt& a);
Use std::deque container to store digits of BigInt.

Problem 06 (object decomposition) (4%)

Solve UVA problem 10523 “Very Easy” using class BigInt from previous problem.
Add operator:
    BigInt operator*(const BigInt& a, const BigInt& b).

Problem 07 (object decomposition) (4%)

Solve UVA problem 10494 “If We Were a Child Again” using class BigInt from previous problem.
Add operators:
  BigInt operator/(const BigInt& a, const BigInt& b);
  BigInt operator%(const BigInt& a, const BigInt& b);