ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ
Κ10: ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΑΦΗΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ
Εξετάσεις 1 Οκτωβρίου 2007

1. (α′) Δώστε δύο παραδείγματα προβλημάτων προς υλοποίηση για τα οποία ενδείκνυται κάποιος να χρησιμοποιήσει C και δύο για τα οποία ενδείκνυται κάποιος να χρησιμοποιήσει C++.

   (β′) Έστω ότι έχουμε το παρακάτω C++ πρόγραμμα:

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

class A{ int data;
  public:
   A(int dat=0) : data(dat) { cout << "I just constructed an A" << endl; }
   VVVV void print() { cout << "The A data is: " << data << endl; } };

class B : public A{ int data;
  public:
   B(int dat=0) : data(dat) { cout << "I just constructed a B" << endl; }
   void print() { A::print(); cout << "The B data is: " << data << endl; } };

main(){ A a(10); B b(5); A* pa = &a; B* pb = &b;
        a.print(); b.print(); pa->print(); pb->print();
        pa = pb; pa->print(); a = b; a.print(); }
```

   Ποιό είναι το αποτέλεσμα της εκτέλεσής του στην περίπτωση που αντικαταστήσουμε το VVVV με το κενό και ποιο στην περίπτωση που το αντικαταστήσουμε με τη λέξη virtual; Αιτιολογήστε την απάντησή σας.

2. Αιτιολογώντας την απάντησή σας, δώστε το αποτέλεσμα της εκτέλεσης του παρακάτω προγράμματος C++:

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

class Block{
   int data;
  public:
   Block(int dat=0) : data(dat)
      { cout << "I just constructed a Block" << endl;}
   void inc() { data++; }
   void print() { cout << "The Block data is: " << data << endl; } };

class B1{
   Block b1; Block& b2; Block b3;
  public:
   B1(Block& blk1, Block& blk2, Block& blk3) :
      b1(blk1), b2(blk2), b3(blk3)
      { cout << "I just constructed a B1" << endl;}
   void inc() { b1.inc(); b2.inc(); b3.inc(); }
   void print() { b1.print(); b2.print(); b3.print(); } };
```

```
class B2{
   Block* pb1; Block* pb2; Block* pb3;
  public:
   B2(Block* pblk1 = NULL, Block* pblk2 = NULL, Block* pblk3 = NULL) :
      pb1(pblk1), pb2(pblk2), pb3(pblk3)
      { cout << "I just constructed a B2" << endl;}
   void inc() { if (pb1 != NULL) pb1->inc();
                if (pb2 != NULL) pb2->inc();
                if (pb3 != NULL) pb3->inc(); }
   void print() { if (pb1 != NULL) pb1->print();
                  if (pb2 != NULL) pb2->print();
                  if (pb3 != NULL) pb3->print(); } };
class B{
   B1 b1; B2 b2;
  public :
   B(B1& db1, B2& db2) : b1(db1), b2(db2)
     { cout << "I just constructed a B" <<endl;}
   void inc() { b1.inc(); b2.inc(); }
   void print() { b1.print(); b2.print(); }
   B tricky() { inc(); return *this; } };

main(){
   Block block(10); B1 b1(block, block, block);
   B2 b2(&block, &block, &block); B b(b1,b2);
   block.print(); b1.print(); b2.print(); b.print();

   block.inc(); b1.inc(); b2.inc(); b.inc();
   block.print(); b1.print(); b2.print(); b.print();

   b.tricky().inc();
   block.print(); b1.print(); b2.print(); b.print(); }
```

3. Έστω ότι έχουμε το παρακάτω C++ πρόγραμμα:

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

class Athlete{
  protected: int speed;
  public:
   Athlete(int spd=0) : speed(spd) {}
   virtual void run() = 0;
   void hit() { cout << "I was hit" << endl; stop(); } ;
   virtual void stop() { if ( speed !=0 ) {speed = 0; cout << "I just stopped";} } };

class StrongAthlete : public Athlete{
  public:
   StrongAthlete(int spd) : Athlete(spd) {}
   void run() { speed = speed + 30; cout << " I run fast" << endl; }
   void hit() { cout << "Nothing happens" << endl; } };
```

```
class HeavyAthlete : public StrongAthlete{
  public:
   HeavyAthlete(int spd) : StrongAthlete(spd) {}
   void run() { speed = speed + 10; cout << " I run not so fast" << endl; }
   void stop() { if ( speed!=0 ) speed = speed - 10;
                 cout << "I cannot stop" << endl; } };

class DelicateAthlete : public Athlete{
  public:
   DelicateAthlete(int spd=0) : Athlete(spd) {}
   void run() { speed = speed + 20; cout << " I am agile" << endl; } };

void train(AAAA what) { what.run(); what.hit(); }

main(){ BBBB what(10); train(what); }
```

Με τι μπορείτε να αντικαταστήσετε τα `AAAA` και `BBBB`; Ποιο είναι το αποτέλεσμα της εκτέλεσης για τον κάθε συνδυασμό αντικαταστάσεων; Αιτιολογήστε την απάντησή σας (Σημείωση: όπου είναι απαραίτητο, στη συνάρτηση `train`, αντικαταστήστε το `.` με `->`. Επίσης, όπου είναι απαραίτητο, στη συνάρτηση `main`, αντικαταστήστε το `what` με `&what`.)

4. Να υλοποιηθεί σε C++ μια προσομοίωση νεανικού κέντρου διασκέδασης. Το κέντρο διασκέδασης αποτελείται από 100 θέσεις καθισμένων και M όρθιων (είναι δηλαδή συνολικής χωρητικότητας $100 + M$ πελατών). Επίσης, σερβίρονται τέσσερα ποτά. Για το κάθε ποτό, υπάρχει μια αρχική ποσότητα κατανεμημένη ισομερώς σε τρεις μπάρες. Ένας πελάτης αναπαρίσταται από την επιλογή του σε ποτό. Για να μπει στο κέντρο, αρκεί το κέντρο να μην είναι πλήρες και να υπάρχει διαθέσιμο απόθεμα από το ποτό της επιλογής του. Όταν μπει, προστίθεται στην ουρά αναμονής της μπάρας με τη μικρότερη ουρά. Μια εξυπηρέτηση πελάτη γίνεται από τη μπάρα με τη μεγαλύτερη ουρά αναμονής και μειώνει την διαθέσιμη ποσότητα ποτού της επιλογής του κατά μια μονάδα. Κατόπιν, αν υπάρχουν διαθέσιμες θέσεις καθισμένων, ο πελάτης αυτός καταλαμβάνει μια από αυτές αλλιώς μένει όρθιος μειώνοντας τις διαθέσιμες θέσεις όρθιων κατά μία. Η έξοδος πελατών δεν ενδιαφέρει την προσομοίωση. Για την προσομοίωση να χρησιμοποιηθούν οι κλάσεις "Νεανικό Κέντρο Διασκέδασης" και "Μπάρα".

   Η κλάση "Νεανικό Κέντρο Διασκέδασης"

   - έχει τρεις μπάρες (`bars`)
   - έχει τη διαθεσιμότητα τεσσάρων ποτών (`drinks`)
   - έχει μια χωρητικότητα όρθιων (`standing_capacity`)
   - έχει ένα σύνολο 100 θέσεων με ενδεχόμενα καθισμένους πελάτες (`sitting`)
   - έχει έναν αριθμό που αντιστοιχεί στους όρθιους στο κέντρο (`standing`)

   Η κλάση "Νεανικό Κέντρο Διασκέδασης" χαρακτηρίζεται από την εξής συμπεριφορά:

   - αρχικά, στις μπάρες δεν περιμένει κανείς, δεν κάθεται κανείς και δεν είναι όρθιος κανείς. Επίσης, προσδιορίζεται η διαθεσιμότητα των τεσσάρων ποτών που μοιράζεται ισομερώς στις τρεις μπάρες.
   - ένας πελάτης εισέρχεται στο κέντρο, αν αυτό είναι επιτρεπτό, και προστίθεται στην ουρά της μπάρας με τη μικρότερη ουρά αναμονής (σε περίπτωση που οι μπάρες με ουρές ελαχίστου μήκους είναι περισσότερες της μίας, επιλέγεται μια οποιαδήποτε μπάρα από αυτές).
   - γίνεται εξυπηρέτηση, εξυπηρετώντας έναν πελάτη από τη μπάρα με τη μεγαλύτερη ουρά αναμονής, βάζοντάς τον να καταλάβει μια θέση καθισμένων, αν υπάρχει διαθέσιμη (σε περίπτωση που οι μπάρες με ουρές μεγίστου μήκους είναι περισσότερες της μίας, επιλέγεται μια οποιαδήποτε μπάρα από αυτές). Σε περίπτωση που τέτοια θέση δεν υπάρχει, τον προσθέτει στον αριθμό των όρθιων.

- η διαθεσιμότητες του κέντρου σε ποτά προκύπτουν από τις αντίστοιχες διαθεσιμότητες από τις τρεις μπάρες.

Η κλάση "Μπάρα"

- έχει τη διαθεσιμότητα των τεσσάρων ποτών (`drinks`)
- έχει μια ουρά πελατών που περιμένουν να εξυπηρετηθούν (`waiting_queue`)
- έχει το πλήθος των πελατών που περιμένουν να εξυπηρετηθούν (`waiting`)

Η κλάση "Μπάρα" χαρακτηρίζεται από την εξής συμπεριφορά:

- κατά την κατασκευή της, δεν περιμένει κανείς και προσδιορίζεται η διαθεσιμότητα των τεσσάρων ποτών.
- ένας νέος πελάτης προστίθεται να εξυπηρετηθεί, περιμένοντας στο τέλος των πελατών που περιμένουν να εξυπηρετηθούν.
- ο πρώτος πελάτης που περιμένει να εξυπηρετηθεί, εξυπηρετείται μειώνοντας τη διαθεσιμότητα του ποτού της επιλογής του κατά μια μονάδα. Επίσης, αφαιρείται από την ουρά αναμονής των πελατών. Σημείωση: η διαθεσιμότητα ενός ποτού σε μια μπάρα μπορεί να πάρει και αρνητική τιμή.

Να υλοποιήσετε σε C++ τις παραπάνω κλάσεις, ορίζοντας νέες όπου δείχνει ενδεικνυόμενο.
Σημείωση: Όπου θεωρείτε απαραίτητο, κάνετε παραδοχές στις προδιαγραφές, τεκμηριώνοντάς τις.