ЕЛЕКТРОТЕХНИЧКИ ФАКУЛТЕТ
Одсек за СИ Београд 09.12.2016.

Други колоквијум из
Објектно оријентисаног програмирања I

1) *(30 поена)*

а) Да ли је могуће приликом преклапања операторске функције променити (у односу на одговарајући оператор за уграђене типове података): (1) тип резултата (2) број операнада (3) смер груписања (4) приоритет? Ако постоји изузетак у односу на начелно правило, навести га.

б) Да ли се из метода приватно изведене класе може приступити заштићеном члану основне класе и зашто?

в) Који модификатор, и на ком месту у декларацији метода треба да стоји, тако да: (1) омогући надјачавање метода (полиморфну редефиницију) у потклаcи; (2) забрани надјачавање метода у потклаcи; (3) нагласи да је метод надјачавање метода из основне класе; (4) нагласи да метод сакрива истоимене методе из основне класе?

2) *(укупно 70 поена)* Написати на језику *C++* следеће класе (класе опремити оним конструкторима, деструктором и операторима доделе који су потребни за безбедно и ефикасно коришћење класа):

- *(15 поена)* ***Налепница*** има назив и једнозначни аутоматски генерисан идентификатор који може да се дохвати. Назив се задаје приликом стварања. При копирању и додели вредности мора се обезбедити једнозначност идентификатора. На главном излазу се исписује (`it<<nalepnica`) у облику *назив:идентификатор*.
- *(25 поена)* ***Флаша*** пића има налепницу, цену без кауције (у коју није урачуната цена флаше), запремину (подразумевано 0.33 литара) и врсту пића. Врста пића може бити `BEZALKOHOLNO` (подразумевано) и `ALKOHOLNO`. Сви подаци се задају приликом стварања и могу да се дохвате, а једино цена може да се промени. Може да се израчуна цена флаше пића са кауцијом. Могуће је проверити да ли су две флаше пића једнаке (`flasa1==flasa2`). Флаше су једнаке уколико имају исти назив налепнице, исту цену са кауцијом и врсту пића. На главном излазу се исписује (`it<<flasa`) у облику *налепница* **(***цена_без_кауције***,** *запремина***,** *врста_пића***)**.
- ***Стаклена*** флаша пића је флаша пића код које се цена са кауцијом рачуна тако што се на цену без кауције дода 5% уколико је запремина мања од 0.5 литара или 10% уколико је запремина једнака или већа од 0.5 литара. На главном излазу се исписује (`it<<staklena`) у облику *флаша***`-staklena`**.
- ***Пластична*** флаша пића је флаша пића код које је цена са кауцијом једнака цени без кауције. На главном излазу се исписује (`it<<plasticna`) у облику *флаша***`-plasticna`**.
- *(20 поена)* ***Дисконт*** има назив и садржи произвољан број флаша. Ствара се празан, после чега се флаше пића додају појединачно (`diskont+=flasa`), те дисконт постаје њихов власник. Приликом уништавања дисконта, уништавају се и флаше пића које се налазе у њему. Може да се дохвати флаша пића по задатом целобројном индексу (`diskont[indeks]`). Може да се израчуна укупна вредност свих флаша у дисконту, при чему се рачунају цене са кауцијом (`~diskont`). Дисконт не може да се копира ни премешта, ни иницијализацијом ни доделом. На главном излазу се исписује (`it<<diskont`) тако што се испише назив дисконта, а потом у засебним редовима испишу садржане флаше пића.

(10 поена) Написати на језику *C++* **програм** који створи један дисконт задатог назива и у њега дода неколико флаша пића, испише га на главном излазу, а затим у новом реду испише укупну цену са кауцијом свих флаша у дисконту. Није потребно ништа учитати с главног улаза.

НАПОМЕНЕ: **а)** Колоквијум траје **120** минута.

б) Рад се предаје искључиво у вежбанци за испите (-5 поена за неадекватну вежбанку). Није дозвољено имати поред себе друге листове папира, нити уз себе имати мобилни телефон, без обзира да ли је укључен или искључен.

в) Водити рачуна о уредности. Нечитки делови текста ће бити третирани као непостојећи. Решења задатака навести по горњем редоследу (-1 поен за лош редослед). Препоручује се рад обичном графитном оловком.

г) Решење задатка не треба раздвајати у датотеке. Довољно је за сваку класу навести дефиницију класе и одмах иза ње евентуалне дефиниције метода које нису дефинисане у самој класи.

д) Резултати колоквијума биће објављени на *Web*-у на адреси:
`http://rti.etf.bg.ac.rs/rti/ir2oo1/index.html/`

```
#include <string>
#include <iostream>
using namespace std;
class Nalepnica {
public:
  Nalepnica(string n) { naziv = n; }
  Nalepnica(const Nalepnica& np) {
    naziv = np.naziv;
  }
  Nalepnica& operator=(const Nalepnica& np){
    if (this != &np) { naziv = np.naziv; }
    return *this;
  }
  friend ostream& operator<<(ostream& it,
                       const Nalepnica& n) {
    return it << n.naziv << ": " << n.id;
  }
  int dohvId() const { return id; }
  string dohvNz() const { return naziv; }
private:
  static int ID; int id=++ID; string naziv;
};
int Nalepnica::ID = 0;
```

```
class Flasa {
public:
  enum Vrsta { BEZ, ALK };
  Flasa(Nalepnica& n,double c,double z=0.33,
                  Vrsta v = BEZ) : nlp(n) {
    cena = c; zap = z; vrsta = v;
  }
  virtual ~Flasa() {}
  double dohvCena() const { return cena; }
  double dohvZap() const { return zap; }
  const Nalepnica& dohvNlp() const { return
                                       nlp; }
  virtual double cenaSaKauc() const = 0;
  void promeniCenu(double c) { cena = c; }
  friend bool operator==(const Flasa& f1,
                         const Flasa& f2){
    return f1.nlp.dohvNz()==f1.nlp.dohvNz()
     && f1.cenaSaKauc() == f2.cenaSaKauc()
     && f1.vrsta == f2.vrsta;
  }
  friend ostream& operator<<(ostream& it,
                         const Flasa& f) {
    f.pisi(it); return it;
  }
protected:
   Nalepnica& nlp;Vrsta vrsta;double cena,zap;
   virtual void pisi(ostream& it) const {
    it << nlp << "(" << cena << ", " << zap
                                    << ", ";
    if (vrsta == Vrsta::BEZ)
      it << "bezalkoholno";
    else it << "alkoholno";
    it << ")";
  }
};
```

```
class Staklena : public Flasa {
public:
  Staklena(Nalepnica& n, double c,
            double z = 0.33, Vrsta v = BEZ)
                    : Flasa(n, c, z, v) { }
  double cenaSaKauc() const override {
    if (zap <= 0.5) return cena * 1.05;
    else return cena * 1.10;
  }
private:
  void pisi(ostream& it) const override{
    Flasa::pisi(it); it << "-staklena";
  }
};
```

```
class Plasticna : public Flasa {
public:
  Plasticna(Nalepnica& n, double c,
            double z = 0.33, Vrsta v = BEZ)
                    : Flasa(n, c, z, v) { }
  double cenaSaKauc() const override {
    return cena; }
private:
  void pisi(ostream& it) const override{
    Flasa::pisi(it); it << "-plasticna";
  }
};
```

```
class DiskontPica {
public:
  DiskontPica(string i) { ime = i;
                    prvi = posl = nullptr; }
  DiskontPica(const DiskontPica&) = delete;
  DiskontPica& operator=(const DiskontPica&)
                                    = delete;
  Flasa* operator[](int);
  const Flasa* operator[](int) const;
  void operator+=(Flasa*);
  double operator~() const;
  friend ostream& operator<<(ostream&,
                         const DiskontPica&);
private: string ime;
  struct Elem { Flasa* f; Elem *sled;
    Elem(Flasa* ff, Elem* s = nullptr):
                              f(ff), sled(s) {}
    ~Elem() { delete f; }
  };
  Elem *prvi, *posl;
  void brisi();
};
void DiskontPica::brisi() {
  while (prvi) { Elem* stari = prvi;
    prvi = prvi->sled; delete stari;
  }
  posl = nullptr;
}
Flasa* DiskontPica::operator[](int i) {
  Elem *tek = prvi;
  while (i-- > 0) tek = tek->sled;
  return tek->f;
}
const Flasa* DiskontPica::operator[](int i)
                                      const {
  return const_cast<const Flasa*>
     ((const_cast<DiskontPica&>(*this))[i]);
}
void DiskontPica::operator+=(Flasa* fp) {
  posl = (!prvi ? prvi : posl->sled) =
                                 new Elem(fp);
}
double DiskontPica::operator~() const {
  double s = 0; Elem *tek = prvi;
  while (tek) { s += tek->f->cenaSaKauc();
    tek = tek->sled;
  }
  return s;
}
ostream& operator<<(ostream& it,
                      const DiskontPica& dp) {
  it << dp.ime << endl;
  DiskontPica::Elem *tek = dp.prvi;
  while (tek) { it << *tek->f << endl;
    tek = tek->sled;
  }
  return it;
}
```

```
int main() {
  DiskontPica d("Podrum");
  Nalepnica n1("Sok"),n2("Vino"),n3("Pivo");
  d += new Staklena(n1, 60, 0.5);
  d += new Staklena(n2,1100,0.7,Flasa::ALK);
  d += new Plasticna(n3,150,2,Flasa::ALK);
  cout << d << endl;
  cout << "Ukupna vrednost: " << ~d << endl;
}
```

```
Podrum
Sok: 6(60, 0.5, bezalkoholno)-staklena
Vino: 9(1100, 0.7, alkoholno)-staklena
Pivo: 12(150, 2, alkoholno)-plasticna

Ukupna vrednost: 1423
```