ЕЛЕКТРОТЕХНИЧКИ ФАКУЛТЕТ
*Одсек за СИ*

Београд 25.10.2013.

## Први колоквијум из
# Објектно оријентисаног програмирања I

**1)** *(30 поена)* Одговорити концизно (по једна или две реченице) и прецизно на следећа питања:

**а)** Који ефекат производи оператор `new`?

**б)** Шта представља показивач `this`, којег је типа и да ли се може користити у статичкој функцији чланици неке класе?

**в)** Под којим условима конструктор копије може да има више параметара (формалних аргумената)?

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

- *(25 поена)* ***Време*** се задаје помоћу броја дана, месеца, године, часова минута и секунди као два дугачка цела броја по шеми `ggggmmdd` и `ccmmss` (подразумевано 25.10.2013. у 19:00:00 – исправност не треба проверавати). Време Може да се упореди с другим временом (резултат је <0, =0 или >0, зависно од тога да ли је текуће време пре, једнак или после задатог времена) и да се испише на главном излазу у облику *дан*`.`*мес*`.`*год*`.(`*час*`:`*мин*`:`*сек*`)`.
- ***Догађај*** има задато време догађања и опис догађаја (ниска) који могу да се дохвате. Догађај може да се испише на главном излазу у облику *време*`|`*опис*.
- *(35 поена)* ***Дневник*** има задато име власника (ниска) и може да садржи произвољан број догађаја уређених по неопадајућем редоследу времена догађања. Ствара се празан, после чега догађаји се додају појединачно. Може да се на главном излазу испише име власника дневника, а у наредним редовима догађаји записани у двенику.

*(10 поена)* Написати на језику *C++* **програм** који направи један дневник, дода неколико догађаја и испише дневник на главном излазу. Користити фиксне параметре – није потребно ништа учитати с главног улаза.

---

**НАПОМЕНЕ:** **а)** Колоквијум траје **100** 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: `home.etf.rs/~kraus/` (одреднице: *настава* | <име предмета> | *оцене* | *колоквијуми*).

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

class Vreme {
  long dat, kad;
public:
  Vreme(long d=20131025,
        long k=190000)
    { dat = d; kad = k; }
  int uporedi(const Vreme& vre)const {
    if (dat != vre.dat)
      return dat - vre.dat;
    return kad - vre.kad;
  }
  void pisi() const {
    cout << dat%100 << '.'
         << dat/100%100 << '.'
         << dat/10000 << ".("
         << kad/10000 << ':'
         << kad/100%100 << ':'
         << kad%100 << ')';
  }
};
```

```
class Dogadjaj {
  Vreme vreme;
  char* opis;
public:
  Dogadjaj(Vreme vr, const char* op) {
    opis = new char [strlen(op)+1];
    strcpy(opis, op);
    vreme = vr;
  }
  Dogadjaj(const Dogadjaj& dg) {
    opis =
      new char [strlen(dg.opis)+1];
    strcpy(opis, dg.opis);
    vreme = dg.vreme;
  }
  ~Dogadjaj() { delete [] opis; }
  Vreme dohvVreme() const
    { return vreme; }
  const char* dohvOpis() const
    { return opis; }
  void pisi() const {
    vreme.pisi(); cout << '|' << opis;
  }
};
```

```
class Dnevnik {
  char* vlasnik;
  struct Elem {
    Dogadjaj dog;
    Elem* sled;
    Elem(const Dogadjaj& dg,
         Elem* s=0): dog(dg)
      { sled = s; }
  };
  Elem* prvi;
public:
  explicit Dnevnik(const char* vl) {
    vlasnik = new char [strlen(vl)+1];
    strcpy(vlasnik, vl);
    prvi = 0;
  }
  Dnevnik(const Dnevnik& dn);
  ~Dnevnik();
  Dnevnik& stavi(const Dogadjaj& dg);
  void pisi() const;
};

Dnevnik::Dnevnik(const Dnevnik& dn) {
  vlasnik =
    new char [strlen(dn.vlasnik)+1];
  strcpy(vlasnik, dn.vlasnik);
  prvi = 0;
  for (Elem *tek=dn.prvi, *posl=0;
       tek; tek=tek->sled)
    posl = (!prvi ? prvi : posl->sled)
         = new Elem(tek->dog);
}

Dnevnik::~Dnevnik() {
  delete [] vlasnik;
  while (prvi) { Elem* stari = prvi;
    prvi = prvi->sled; delete stari; }
}

Dnevnik& Dnevnik::stavi(const
Dogadjaj& dg) {
  Elem *tek = prvi, *preth = 0;
  while (tek && tek->dog.dohvVreme()
        .uporedi(dg.dohvVreme())<=0)
    { preth = tek; tek = tek->sled; }
  Elem* novi = new Elem(dg, tek);
  (!preth ? prvi : preth->sled)=novi;
  return *this;
}

void Dnevnik::pisi() const {
  cout << vlasnik << endl;
  for (Elem* tek=prvi; tek;
       tek=tek->sled)
    { tek->dog.pisi(); cout << endl; }
}
```

```
int main() {
  Dnevnik d("Marko");
  d.stavi(Dogadjaj(
     Vreme(20131013, 85530),"Kisa"))
   .stavi(Dogadjaj(
     Vreme(),"Kolokvijum"))
   .stavi(Dogadjaj(
     Vreme(20131020, 151515),"Sunce"))
   .pisi();
}
```

```
Marko
13.10.2013.(8:55:30)|Kisa
20.10.2013.(15:15:15)|Sunce
25.10.2013.(19:0:0)|Kolokvijum
```