AAAAAAAAAAAAAAAAAAAAAAAAAAAA

Programozás alapjai II. 4. ellenőrző dolgozat. 2012.04.19. Kurz/**Terem**: G3/	**20 perc**
Név: **Neptun:**	Összpont:

1. feladat 4 pont

Egy **naplózó rendszerben** (*Log)* különböző eseményeket (*Event*) szeretnénk tárolni. Tudjuk, hogy legfeljebb 100 esemény lehet. Tárolni kell az esemény idejét és további adatokat, melyek teljesen eltérőek lehetnek. Keletkezhet pl. hőfokot (*Temp*) vagy szöveges üzenetet (*Message*) tartalmazó esemény is, de nem zárható ki, hogy újabb események is lesznek. A hőfokot valós számként (*double*), az üzenetet pedig *String*-ként kell tárolni. Az idő tárolására a *Time* osztályt kell használni, melynek az alapértelmezett konstruktora mindig az aktuális időt tárolja el. A rendszerben megvalósítandó műveleteket egy kódrészlettel mutatjuk be:

```
Log events;                          // Ez lesz az eseménytár
events.add(new Temp(12.3));          // Hőmérséklet esemény hozzáadása
events.add(new Message("ZH"));       // Szöveges esemény hozzáadása
events.print();                      // események kiírása keletkezési sorrendben
events.clear();                      // Összes esemény törlése
```

Feltételezheti, hogy a *Log* osztályból példányosított objektumot nem akarjuk paraméterként átadni és értékadás jobb, ill. bal oldalán sem szerepel. **Tervezzen** meg olyan osztályhierarchiát, ami alkalmas az események keletkezési sorrendben való tárolására és könnyen bővíthető. Ügyeljen arra, hogy az jól határozza meg az objektumok felelősségét! **Deklarálja** a *Log, Event*, és *Message* osztályokat! **Csak** a *Log* osztály konstruktorát, valamint az *add()* és *print()* metódusát **valósítsa** meg!

```
class Event {
      Time time;
public:
      Event();
      virtual void print();
      virtual ~Event();
};

class Message :public Event {
      String msg;
public:
      Message (String);
      void print();
};
```

```
class Log {
      Event* events[100];
      int numevents;
public:
      Log() :numevents(0) {}
      void add(Event* ev) {
            events[numevents++] = ev;
      }
      void print() {
            for (int i = 0; i < numevents; i++)
                  events[i]->print();
      }
      void clear();
      ~Log();
};
```

2. feladat 2 pont

Tételezze fel, hogy a *Log* osztálynak nincs *print()* metódusa, de van iterátora, amivel egymás után sorban elérhetők a tárolt események. Mutassa be, egy kódrészleten, hogy ezzel az iterátorral hogyan tudná a *print()* metódus funkcióját kiváltani. !

```
for (Log::iterator i = events.begin(); i != events.end(); ++i)
      (*i)->print();
```