AAAAAAAAAAAAAAAAAA rendelkezésre álló idő: **15 perc**

Szoftver laboratórium 2.	4. Ellenőrző dolgozat.	2014.04.14.	Kurz/**Terem**: L1/
Név:	**Neptun:**		Összpont:

Tételezze fel, hogy rendelkezésére áll az előző laboron elkészített *Tartozek* és *Printer* osztály:

```
class Tartozek {
   String gySzam;
   String megnevezes;
public:
   Tartozek(const char*, const char*);
   virtual void print(std::ostream&)
const;
   virtual ~Tartozek();
};
```

```
class Printer : public Tartozek {
   String fajta;
   int sebesseg;
public:
   Printer(const char*, const char*,
                          const char*, int);
   void print(std::ostream&) const;
};
```

1. feladat 2 pont

A *Printer* osztály felhasználásával, annak módosítása nélkül hozzon létre egy olyan hálózati nyomtató (*NetPrinter)* osztályt, ami együtt tárolható a már meglevő tartozékokkal! Az új osztály tárolja a hálózati interfész sebességét is (*double*)! Valósítsa meg az új osztály minden szükséges tagfüggvényét úgy, hogy a *print* tagfüggvény a hálózati interfész sebességét is írja ki a printer egyéb jellemzőivel (gyári szám, megnevezés, …) együtt!

2. feladat 2 pont

Alakítsa át a *NetPrinter* osztályt sablonná úgy, hogy a hálózati interfész sebességének típusa tetszőleges típus lehessen! Elegendő, ha csak a sablon lényegi sorait írja le, és a 1. feladat megoldásával azonos sorokra … -tal hivatkozik! Adja meg, hogy a megoldás milyen minimum követelményeket támaszt a sablonban használt generikus típussal szemben (pl. , van ++ operátora)! Ne írjon olyan követelményt, amit a megoldás nem követel meg!

3. feladat 2 pont

Mutassa be az elkészített sablon használatát egy programrészlettel! Ebben hozzon létre legalább két olyan *NetPrinter* objektumot, amiben eltérő típus tárolja az interfész sebességét! Az objektumokat tárolja heterogén gyűjteményként (megfelelő tömb), majd a gyűjtemény elemeit ciklussal végigjárva írja ki a szabványos kimenetre a létrehozott objektumok adatait!

```
class NetPrinter : public Printer {
   double ifSebesseg;
public:
   NetPrinter(const char* gysz, const char* megn,
              const char* fajta, int seb, double ifseb)
      : Printer(gysz, megn, fajta, seb), ifSebesseg(ifseb) {}
   void print(std::ostream& os) const {
      Printer::print(os);
      os << " interfesz: " << ifSebesseg;
   }
};

template<class T>
class NetPrinter : public Printer {
   T ifSebesseg;
public:
   NetPrinter(const char* gysz, const char* megn,
              const char* fajta, int seb, T ifseb)
...
};
```

T-vel szemben támasztott követelmény:

- **legyen másolható,**
- **legyen olyan std::ostream& operator<<(std::ostream&, const T&) operátor, vagy ezzel kompatibilis, amivel kiírható.**

```
Tartozek *t[10];
t[0] = new NetPrinter<int>("123", "Netprinter", "Laser", 20, 10);
t[1] = new NetPrinter<double>("123", "Netprinter", "Laser", 22, 100);
for (int i = 0; i < 2; i++)
   t[i]->print(std::cout);
```