AAAAAAAAAAAAAAAAAAA rendelkezésre álló idő: **15 perc**

Szoftver laboratórium 2.	5. Ellenőrző dolgozat.	2014.04.29.	Kurz/**Terem**: L4/
Név:	**Neptun:**		Összpont:

1. feladat 2.5 pont

Írjon generikus függvényt (***feltolt***), ami egy sorozattároló összes elemét feltölti egy adott konstanssal! Ez a konstans alapértelmezés szerint (default argumentum) az adott generikus típus konstruktora által létrehozott érték legyen! A sablon és a függvény paramétereit, visszatérési értékét Önnek kell meghatároznia úgy, hogy a 2. feladatban azt hatékonyan fel tudja használni!

```
template <typename Iterator, typename T>
void felolt(Iterator first, Iterator last, const T& value = T()) {
    while(first != last)
        * first++ = value;
}
```

---

2. feladat 3.5 pont

Tételezze fel, hogy rendelkezésére áll egy generikus tároló (pl. a korábbi laborgyakorlatokon elkészített *Array<T>*)! Mutassa be ennek felhasználásával az elkészített sablon használatát a következő kódrészletek megírásával:

1. Hozzon létre 1 db *valós*, 1 db *std::string* és 1 db *size_t* elemeket tartalmazó, azonos méretű tárolót!
2. Az 1. feladatban elkészített generikus algoritmus felhasználásával töltse fel a valós tárolót 123.52-vel!
3. A generikus algoritmus felhasználásával töltse fel az *std::string* elemeket tároló tárolót az Ön Neptun kódjával!
4. Készítsen iterátorra épülő ciklust, ami a *size_t* elemeket tároló tárolót feltölti a stringek hosszával. Tételezze fel, hogy a stringek tárolója megváltozott, de a tárolt elemek száma nem! Feltételezheti továbbá, hogy létezik a *size_t std::string::length()* függvény, a mi megadja egy string hosszát!
5. Tételezze fel, hogy rendelkezésére áll a korábbi laborgyakorlaton elkészített forEach<Iter, Fn> függvénysablon is! Ennek felhasználásával írja a ki *size_t* elemeket tároló tároló tartalmát a szabványos kimenetre! (Készítsen hozzá megfelelő függvényt, vagy függvényobjektumot!)

```
void sizePrint(const size_t len) { std::cout << len << std::endl; }

Array<double> dblTar(20);
Array<std::string> strTar(20);
Array<size_t> sizeTar(20);

feltolt(dblTar.begin(), dblTar.end(), 123.52);
feltolt(strTar.begin(), strTar.end(), "ABCDEF");

Array<size_t>::iterator leni = sizeTar.begin();
for (Array<std::string>::iterator si = strTar.begin(); si < strTar.end(); ++si)
    *leni++ = si->length();

forEach(sizeTar.begin(), sizeTar.end(), sizePrint);
```