5. Gyakorlat

Függvénypointerek, lambda expression, `static` kulcsszó, inline függvények, `friend` kulcsszó

1. A `for_each` függvény használatával írjunk programot, amely egy `Point` típusú objektumokat tartalmazó `vector` esetén kiírja a képernyőre a megelőző pontból az adott pontba mutató vektort. Az első elem esetén a kiindulópont legyen definíció szerint az origó. (Használjuk az előadáson bemutatott `Point` osztályt.)

2. Írjunk C++ függvényt, amely paraméterül kap egy `int`-eket tartalmazó `list`-et és egy további `bound` nevű `int`-et, és kitörli a listából azokat a számokat, amelyek a `bound` értéknél nagyobbak.

3. A következő kódban van egy hiba, keressük meg és javítsuk ki ezt. (Először próbáljuk meg ezt a számítógép használata nélkül megtenni, keressünk több lehetséges megoldást.) Mit ír ki a képernyőre a kijavított program?

```
#include <iostream>
using namespace std;
int main(){
    []() {cout << "Hello" << endl; }();
    int x = [](int x, int y) { return x + y; } (10, 5);
    [x](int y) {cout << "Sum: " << (x += y) << endl; }(5);
}
```

4. Definiáljuk az a_n sorozatot a következőképp: legyen $a_1 = 3$, továbbá $n > 1$ esetén

$$a_n = \prod_{d|n,\, d<n} (a_d + (-1)^d).$$

Írjunk függvényt, amely visszaadja az a_n értéket úgy, hogy minden $n > 0$ esetén a korábban kiszámolt értékeket eltároljuk.

5. Valósítsuk meg a `Graph` osztályt a `map` tároló segítségével a következő lépésekben:

   (a) Definiáljunk egy `Edge` osztályt, amely egy élet reprezentál. Legyen két `unsigned int` tagváltozója, amelyek az élen lévő két csúcs címkéjét jelölik. Valósítsuk meg úgy az osztályt, hogy az élen lévő csúcsokat egy felhasználó lekérdezhesse, de módosítani csak közvetve, a később definiált `Graph` osztályon keresztül tudja. Legyen egy tagfüggvénye, amivel lekérdezhető, hogy hurokélről van-e szó, ill. valósítsuk meg az `==` és `<` operátorokat (utóbbi legyen lexikografikus rendezés a csúcspárokon). Minden tagfüggvény és konstruktor legyen `inline`.

   (b) Definiáljunk egy `Vertex` osztályt, amely egy csúcsot reprezentál. Legyen egy `unsigned int` tagváltozója, amely a csúcs címkéjét jelöli, és tároljuk el a csúcsra illeszkedő éleket egy `Edge` objektumokat tartalmazó listában. Úgy valósítsuk meg az osztályt, hogy egy külső felhasználó a tagváltozók értékéhez hozzáférhessen, de azokat módosítani csak közvetve, a később definiált `Graph` osztályon keresztül tudja. Legyen az osztálynak egy publikus default konstruktora, és legyen egy olyan konstruktor is, aminek a paramétere a csúcs címkéje. Legyen továbbá egy olyan tagfüggvénye, amely visszaadja a csúcsra illeszkedő élek számát. Minden tagfüggvény és konstruktor legyen `inline`.

   (c) Definiáljuk a `Graph` osztályt, amely egy irányítatlan gráf éllistákkal való tárolására alkalmas. A csúcsokat egy `map` tárolóban tároljuk, ahol a kulcs `unsigned int`, míg az érték `Vertex` típusú. Írjunk tagfüggvényeket, melyekkel csúcsokat és éleket adhatunk hozzá és törölhetünk, ill. amelyek visszaadják a gráf csúcs- ill. élszámát, valamint egy megadott csúcs fokszámát és a szomszédainak listáját. Írjunk olyan tagfüggvényt, amely visszaadja, hogy két megadott csúcs közt fut-e él. Legyen olyan tagfüggvény is, amely eldönti, hogy egyszerű-e a gráf, valamint legyen lehetőség a gráfból az élek összehúzásával és a hurokélek törlésével egyszerű gráfot csinálni.