2. Gyakorlat

Konverzió, fejállomány és forrásfájl, tömbök, C string-ek, pointerek, dinamikus memóriafoglalás

1. Mit fog kiírni a képernyőre a következő program? Próbáljuk meg meghatározni először a program futtatása nélkül! Magyarázzuk meg az eredményt.

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

int main() {
 cout << (4 / 3) * 10 << endl;
 cout << 4 * (10 / 3) << endl;
 cout << (4 * 10) / 3 << endl;
 cout << (4 / 3) * 10. << endl;
 cout << (4. / 3) * 10 << endl;
 return 0;
}
```

2. Írjunk függvényt `atlag` névvel, amely visszaadja a kapott `int` tömb első n elemének átlagát. Használjuk a `static_cast` kulcsszót.

3. Írjunk két függvényt, amelyek kiszámolják az F_n Fibonacci-sorozat k-adik tagját, ahol $F_0 = F_1 = 1$, $n > 1$ esetén pedig $F_n = F_{n-1} + F_{n-2}$. Az egyik függvényben használjunk ciklust, míg a másikban számoljunk rekurzívan. Deklaráljuk a függvényeket a `fibonacci.h` fejállományban, a függvények definícióját pedig adjuk meg egy külön forrásfájlban. Teszteljük a függvényeket, számoljuk ki mindkettővel az F_{20}, F_{30}, F_{40} értékeket. Mit tapasztalunk?

4. Írjunk olyan függvényeket, amelyek közül az egyik egy síkbeli pont Descartes-koordinátáit polárkoordinátákká konvertálja, a másik pedig fordítva, polárkoordinátákból számol Descartes-koordinátákat. Deklaráljuk az `xy2polar` és `polar2xy` nevű függvényeket a `mypolar.h` fejállományban, a függvények definícióját pedig adjuk meg egy külön forrásfájlban. Használjuk a `cmath` fejállományt, és kezeljük azokat az eseteket is, amikor az output nem egyértelmű, ill. mikor az input nem megfelelő.

5. Írjunk függvényt `max` névvel, amely visszaadja a kapott `double` tömb legnagyobb abszolút értékű elemét és az indexét is. A bemenetei:

   - *tomb* a `double` tömb
   - *hossz* a tömb hossza
   - Megoldástól függően még lehetnek paraméterek.

6. Írjunk függvényt, amely kap egy C string-et és egy karaktert (`char`), és visszaadja, hogy az adott karakter hányszor szerepel a string-ben. Módosítsuk úgy a programot, hogy visszaadja a karakter első előfordulásának címét is, ha pedig a karakter nem fordul elő, akkor adja vissza a null pointert.

7. Írjunk függvényt, amely kap egy C string-et, és visszaadja az abban található leggyakoribb karaktert.

8. Írjunk *szelet* névvel függvényt, mely a kapott C string-et egy paraméterként kapott indextől kezdve egy szintén paraméterként kapott indexű karakterig lemásolja, majd ezt az új C stringet visszaadja. A paraméterek:

   - *str* a string, amiből másolunk, `char` tömb
   - *bal* az első index, amit másolunk, sima `int`
   - *jobb* az utolsó index, amit még másolunk, sima `int`

9. Írjunk függvényt, amely kap két C string-et, és az elsőből törli a másodikban előforduló összes karaktert. Például a "kiskutya", "ky" bemenetre "isuta"-ra változtatja az eredeti első string-et.