13. feladat: Mankala játék (30 pont)

A Mancala családba tartozó játékok, amelyeket kavicsokkal és üregekkel játszottak, a legősibbek közül valók. A mini mancala változatában, amely kétszemélyes játék, négy üreg van. Az 1. és a 2. üreg az egyik, a 3. és 4. üreg a másik játékoshoz tartozik. A játék kezdetén az üregekbe véletlenszerűen beraknak legfeljebb 8 kavicsot. A játékosok felváltva lépnek, az 1-es játékos kezd. A soron következő játékos kiválaszt a hozzá tartozó üregek közül egyet, majd kiveszi az abban lévő összes kavicsot. Ezután a kivett kavicsokból egyet eldob, a többit pedig kavicsot szétosztja az üregek között az alábbiak szerint. Az órajárással ellentétes irányban haladva minden érintett üregbe rak egy kavicsot, de kihagyja azt az üreget, amelyből kivette a kavicsokat. Például, ha az 1. üreget választja, amiben 6 kavics van, akkor sorrendben a 2., 3., 4., 2. és 3. üregbe rak egy-egy kavicsot.

A játék akkor ér véget, ha a soron következő játékos nem tud lépni, mert mindkét hozzá tartozó üreg üres. Az a játékos nyer, aki utoljára tudott lépni.

Írj programot (**`mankala.pas, mankala.c, mankala.cpp`**), amely a játékot kezdő 1. játékos nyerő stratégiáját valósítja meg!

Könyvtári műveletek

A **`game`** nevü könyvtárból az alábbi három művelet használható:

- **`Pit`**, egy ***i*** egész szám az argumentuma, a függvényhívás eredménye az *i*-edik üregben lévő kavicsok száma. Ezzel kell lekérdezni a kezdeti játékállást.
- **`MyMove`**, ezzel a művelettel közli az 1. játékos az aktuális lépését. Egyetlen argumentuma a lépésben választott üreg sorszáma, ami 1, vagy 2 lehet. A művelet hatására a 2. játékos azonnal lép (ha tud).
- **`YourMove`**, argumentum nélküli művelet, a 2. játékos utolsó lépését adja (ami 3, vagy 4 lehet).

Ha az aktuális játékos nem tud lépni, akkor a program automatikusan befejeződik.
A **`game`** könyvtár műveletei két szöveges állományt hoznak létre, **`mankala.ki`** és **`mankala.log`** néven. A **`mankala.ki`** első sorában a játékot nyerő játékos sorszáma van. A programod és a könyvtár közötti párbeszédet a **`mankala.log`** tartalmazza.

Kikötések

- A Pascal könyvtár neve: **`game.tpu`**
- A Pascal függvények és eljárások deklarációja:
```
function Pit(i: integer):integer;
procedure MyMove(i:integer);
function YourMove:integer);
```
- A C/C++ könyvtár neve: **`game.obj`**
- C/C++ függvények és eljárások deklarációja (**game.h**):
```
int Pit(int i);
void MyMove(int i);
int YourMove(void);
```
- Programod nem olvashat és nem írhat egyetlen állományt sem.
- A programodat csak olyan bemenetekre tesztelik, amelyre a játékot kezdő 1. játékosnak van nyerő stratégiája, tehát nyerhet, bárhogyan is lép az ellenfele.

Gyakorlás

A **`mankala.be`** szöveges állomány első sorában a kezdeti játékállást kell megadni, tehát négy nemnegatív egész számot, amelyek összege legfeljebb 8.

Pascal programozóknak: programod írd be a következő import direktívát:
```
uses game;
```
C/C++ programozóknak: programodba írd be a következő direktívát:
```
#include 'game.h';
```
Hozd létre a **`mankala.prj`** projektet és add hozzá a **`game.obj`** és **`mankala.c`** (**`mankala.cpp`**) állományokat.

Példa játékmenet (mankala.log tartalma):

```
Pit(1)=1
Pit(2)=1
Pit(3)=2
Pit(4)=4
P1: 1: 0 1 2 4
P2: 3: 0 1 0 5
P1: 2: 0 0 0 5
P2: 4: 2 1 1 0
P1: 2: 2 0 1 0
P2: 3: 2 0 0 0
P1: 1: 0 1 0 0
Normál terminálás
```