Programozás I.

Gyakorlás egydimenziós tömbökkel
Többdimenziós tömbök
Gyakorló feladatok

Hallgatói Tájékoztató

A jelen bemutatóban található adatok, tudnivalók és információk a számonkérendő anyag vázlatát képezik. Ismeretük szükséges, de nem elégséges feltétele a sikeres zárthelyinek, illetve vizsgának.

Sikeres zárthelyihez, illetve vizsgához a jelen bemutató tartalmán felül a kötelező irodalomként megjelölt anyag, a gyakorlatokon szóban, illetve a táblán átadott tudnivalók ismerete, valamint a gyakorlatokon megoldott példák és az otthoni feldolgozás céljából kiadott feladatok önálló megoldásának képessége is szükséges.

Programozás I.

Gyakorlás egydimenziós tömbökkel

Többdimenziós tömbök

Gyakorló feladatok

Gyakorló feladat (Szigetek)

Egy repülőgéppel egyenes vonalban végigrepültünk egy tengerszakasz fölött. Egyenlő távolságonként megmértük, hogy a repülő alatt tenger vagy sziget található-e. (Legalább 10 mérést végeztünk.) Mérési eredményeinket egy tömbben tároltuk el: (1) tenger esetén nulla szerepel a tömbben, (2) sziget esetén a sziget adott helyen mért magassága.

Írjon programot, amely véletlenszerűen előállítja a mérési eredményeket, majd különböző statisztikai feladatokat old meg. Minden részfeladatot különböző metódusokkal valósítson meg!

Gyakorló feladat (Szigetek)

1. Valósíts meg a tömb véletlenszerű feltöltését!
   40% a valószínűsége, hogy egy mérési helyen szigetet találunk. A sziget aktuális magassága 1 és 10 közötti véletlen szám.
2. Jelenítse meg a mérési eredményeket a képernyőn!
3. Határozza meg, hogy hol található (először) a legmagasabb pont, és mennyi ennek a magassága!
4. Adja meg, hogy a legmagasabb pont hányszor fordult elő a repülés során!
5. Határozza meg a leghosszabb szigetszakasz hosszát!
6. Adja meg, hogy a leghosszabb szigeten található-e az első maximális magasságú mérési pont!

Programozás I.

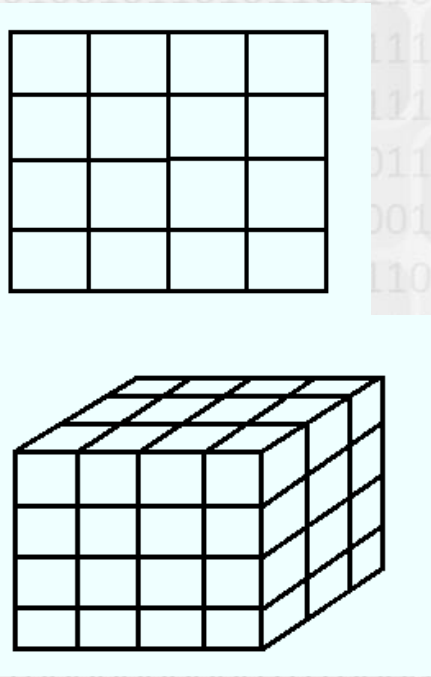
Gyakorlás egydimenziós tömbökkel

Többdimenziós tömbök

Gyakorló feladatok

Többdimenziós tömbök

- **2 dimenziós tömb**
  - sorok és oszlopok
  - elem elérése 2 indexszel
- ***3 dimenziós tömb***
  - *sorok, oszlopok, lapok*
  - *elem elérése 3 indexszel*
- ***N dimenziós tömb***
  - *0., 1., … N. dimenzió*
  - *elem elérése N indexszel*



Többdimenziós tömbök – Deklaráció

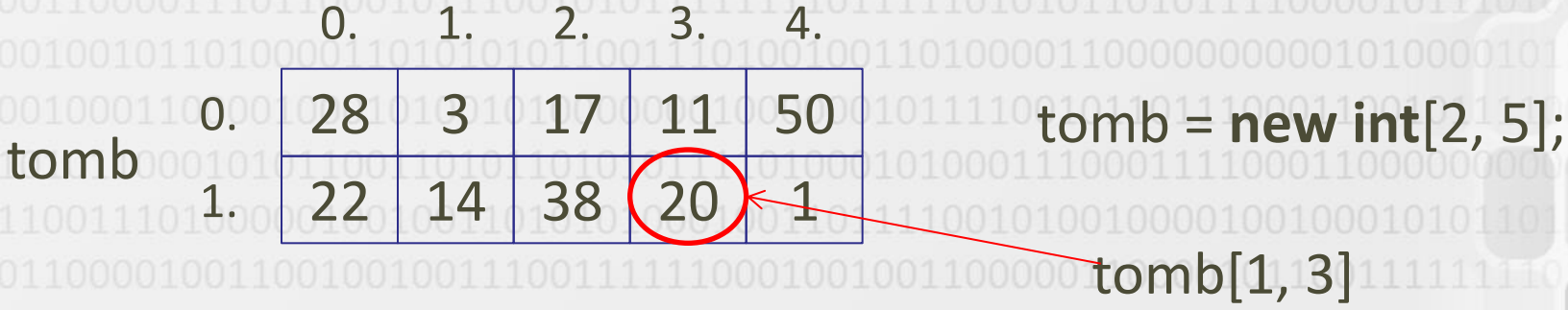
- **Általános formátum:**
  típus*[*vesszők*] *tömbnév;
- **A szögletes zárójelbe dimenziószám-1 darab vesszőt kell tenni**
- **Példák:**
  - **int**[,] matrix;
  - ***bool****[,,] haromdimenziostomb;*
  - ***double****[,,,,] otdimenziostomb;*

Többdimenziós tömbök – Tömblétrehozás

- **Általános formátum:**
  ***tömbnév* = new *tipus* [*elemszám1*, …, *elemszámN*]**
- **Az egyes dimenziók elemszámait vesszőkkel elválasztva kell megadni**
- **A deklaráció és a tömblétrehozás itt is összevonható**
- **Példák**
  - matrix = **new int**[3, 5];
  - haromdimenziostomb = **new bool** [4, 2, 5];
  - **int** t[,,] = **new int**[3, 3, 3];
  - int[,] egeszmatrix = {{1, 2, 3, 4}, {5, 6, 7, 8}, {9, 0, 1, 2}};

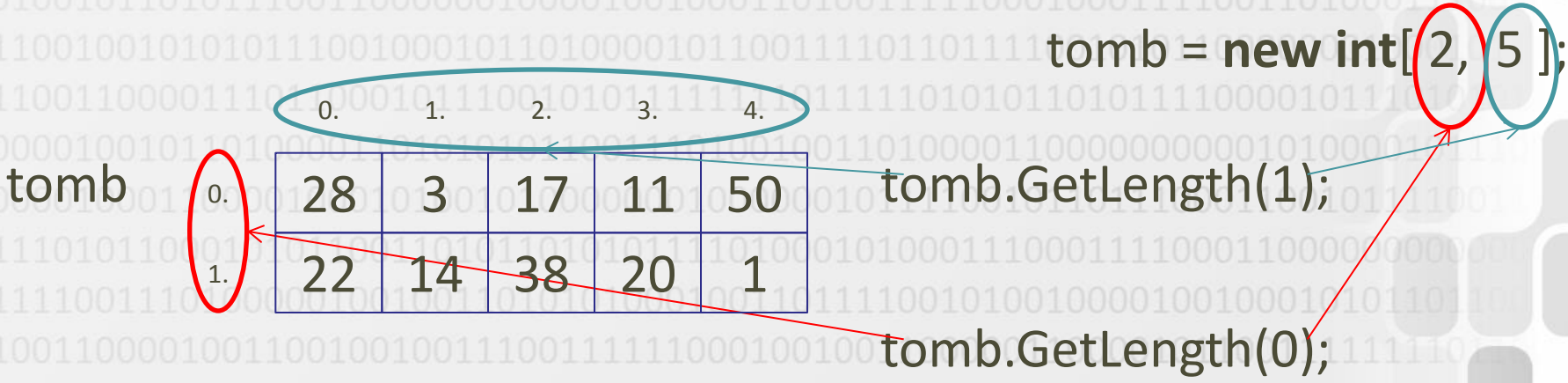
Tömbelem elérése (indexelés)

- **A szögletes zárójelek közé a tömbelem minden egyes dimenzióján belüli sorszámait kell vesszőkkel elválasztva megadni:**
  ***tömbnév*[*index1, index2, …, indexN*]**
- **Az indexekre vonatkozó szabályok u.a., mint az egydimenziós tömbnél**
- **pontosan annyi indexet kell megadni, ahány dimenziós a tömb**



Tömb hosszának (elemei számának) lekérdezése

- **Elemek számának lekérdezése:**
  - Összes tömbben lévő elem darabszáma: *tömbnév*.Length
  - Egy adott dimenzió elemszáma (sorok száma, oszlopok száma, …): *tömbnév*.GetLength(*dimenziósorszám*);



Programozás I.

Gyakorlás egydimenziós tömbökkel

Többdimenziós tömbök

Gyakorló feladatok

Gyakorló feladat (Cserélgetős játék)

Egy játéktábla mezői kétféle módon vannak megjelölve (pl. * és -). Kezdetben minden mező azonos jelölésű (-), kivéve a játéktábla közepén lévő mező, valamint annak négy közvetlen szomszédja.

A játék során a felhasználó megadja a játéktábla egy koordinátáját. A kiválasztott koordinátájú mező, illetve annak négy szomszédja az addigival ellentétes jelölésűre változik.

A játék akkor ér véget, ha a felhasználó minden mezőt *-ra tudott változtatni.

Készítse el a játszást lehetővé tevő programot!

Gyakorló feladat (Cserélgetős játék)

A játéktábla aktuális állapotát egy kétdimenziós logikai tömbben tárolja el!

Megvalósítandó metódusok:

1. `static void init(bool[,] game)`
   A játéktábla kezdeti állapotát előállító metódus
2. `static string state(bool[,] game)`
   A játéktábla aktuális állapotát `string` formában megadó metódus
3. `static void shoot(bool[,] game, int x, int y)`
   Kiválasztott pontra „lövést” megvalósító metódus
4. `static bool isOver(bool[,] game)`
   A metódus vizsgálja, hogy minden mező *-gá vált-e

Gyakorló feladat (Lottó)

Egy szerencsejátékos egy héten több lottószelvénnyel is játszik. Mindegyik szelvényt véletlenszerűen tölt ki.

Készítsünk a játékos számára egy programot, amely „kitölti” számára a szelvényeket, majd a húzást követően kiírja, hogy melyik szelvénnyel hány találatot ért el!

(Az alkalmazás minden lottó típus esetén működjön, a +1 találatok kezelését kivéve!)

Gyakorló feladat (Lottó)

A kitöltött szelvényeket egy kétdimenziós, a kihúzott számokat pedig egy egydimenziós tömbben tárolja el!

Megvalósítandó metódusok:

1. `static int[] GenerateDraw()`
   A kihúzott számokat előállító metódus
2. `static string OutDraw(int[] t)`
   A kihúzott számok megjelenítését támogató metódus
3. `static int[,] GenerateLotto()`
   A szelvényeket „kitöltő” metódus
4. `static string OutLotto(int[,] t, int[] u)`
   A kitöltött szelvények és a találatok számának megjelenítését támogató metódus
5. `static bool Inside(int[] t, int value)`
   A metódus eldönti, hogy egy adott szám szerepel-e egy adott szelvényen

Irodalom, feladatok

- **Kotsis-Légrádi-Nagy-Szénási: Többnyelvű programozástechnika, PANEM, Budapest, 2007**
- **Faraz Rasheed: C# School, Synchron Data, 2006 http://www.programmersheaven.com/2/CSharpBook**
- **Reiter István: C# jegyzet, DevPortal, 2010, http://devportal.hu/content/CSharpjegyzet.aspx**