Imperatív Programozás Labor ZH – 2023.11.17.

Elvárások a programmal szemben
• A megoldáshoz semmilyen segédeszköz nem használható, kivéve a C referenciát.
• A program végleges verziójának működő képesnek kell lennie. Forduljon és fusson!
A nem forduló kód 0 pontot ér!
• Ne használj globális változókat! Csak a makrók megengedettek!
• Törekedj a szép, áttekinthető kódolásra, használj indentálást, kerüld a kódismétlést!
• Logikusan tagold a megoldást!
A kötelező függvényeken túl hozz létre saját függvényeket, ha szükségét látod!
• Kerüld a nem definiált viselkedést okozó utasításokat!
A végleges programban a be nem tartott elvárások pontlevonással járnak!

Egyszerű Tetris játék

A feladatod a Tetris játék leegyszerűsített verziójának elkészítése lesz. A játék random generál 2x2-es „**polyomino**”-kat (azaz **tetris elemek**et), melyeket a játékos által megszabott oszlophoz illesztve leejtünk a játéktérbe. (Részletek lásd. lentebb.) Az elemek lerakása nem időre megy, és nem lehet őket forgatni. Az elemek nem torzulnak a lerakás során és nem is eshetnek vagy fedhetnek át egymáson. A sikeresen elhelyezett elemek pontot érnek. Ebben a játékban nem foglalkozunk azzal, ha egy sor teljesen betelt, nem tűntetjük el a játéktérről. A játéknak vége van, ha az aktuális elemet nem sikerül maradéktalanul elhelyezni a játéktérben (azaz valamelyik szegmense kilóg).

A feladat összesen **50** pontot ér.
Legalább **10** pontot kell gyűjteni a tárgy sikeres teljesítéshez.
A megoldásra **180 perc** áll rendelkezésre.
A végén csak az egy darab forráskódot (**<neptun_kód>.c** állomány) kell feltölteni. (Az utoljára feltöltött megoldást pontozzuk.)

A kötelezően megvalósítandó alprogramok:

Játéktér inicializálása – init_game() (4 pont)

A játékteret reprezentálja egy **10 x 8**–as karaktereket tároló kétdimenziós tömb. (A dimenziókat tároljuk makrókban.) Az aktuális polyomino-t reprezentálja egy **2 x 2**-es karakter tömb. Mind két mátrixot töltsük fel ’ ’ (**SPACE**) karakterekkel.
A függvény megkapja a játékteret és az aktuális polyominot tároló mátrixot paraméterben.

Következő tetrisz elem legenerálása – next_polyomino() (5 pont)

A játék a következő hét tetrisz elemet legyen képes létre hozni:

1: O_ 2: __ 3: OO 4: OO 5: O_ 6: _O 7: OO
O_ OO O_ _O OO OO OO

(**balról-jobbra:** 1x2es álló elem baloldalon; 2x1-es fekvő elem alul; balfelső, jobbfelső, balalsó, jobbalsó „háromszög”; 2x2es négyzet.)

Ahol ’**O**’ az aktuális polyomino egy szegmense, ’_’ pedig üres mező (továbbra is **SPACE**, csak a jobb láthatóság kedvvért szerepel ’_’ az ábrán). Az alprogram meghívásával a 2x2-es mátrixba véletlenszerűen beállítjuk az egyik mintát.
(Tipp: az egyes mintákat nyugodtan „égessük be” a kódba egyesével, a koordinátáik segítségével. Ez nem számít kódismétlésnek. Ezek közül a randomgenerátor használatával kerül kiválasztásra egy minta.)
A függvény megkapja az aktuális polyominot tároló mátrixot paraméterben.

Játék aktuális állapotának kirajzolása – print_game() (5 pont)

A játék vizualizációjáért felelős alprogram. A kirajzolást kezdjük a következő elhelyezendő tetrisz elem feltüntetésével az alapértelmezett kimeneten, majd alatta az aktuális játéktérrel. A játékteret egyértelműen jelezzük valamilyen kerettel (pl.:**'#'**) és a választható oszlopokat 1-től számozva tüntessük fel! A keret nem kell, hogy a játéktér része legyen.
(Tipp: lentebb az ábrán láthatsz egy variációt a kirajzolásra. A sorok számozása, valamint a több fajta karakterből álló keret megrajzolása nem kötelező része a ZH-nak, de ha időd, kedved engedi lehet kreatívoskodni.)
A függvény megkapja a játékteret és az aktuális polyominot tároló mátrixot paraméterben

Játék frissítése – update_game() (26 pont)

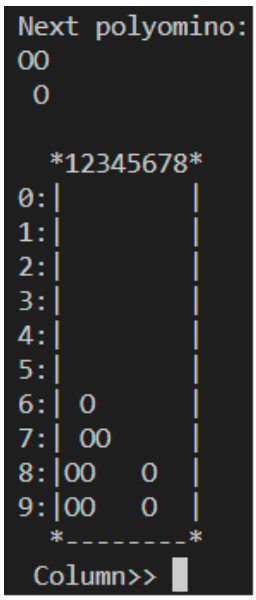
Ez az alprogram a játék „motorja", mely több feladatot is elvégez:

Játék állapotának figyelése (7 pont)
A visszatérésiérték segítségével tájékoztatást ad az elem elhelyezésének sikerességéről: ha a kérdéses oszlopban már nincsen hely**(-2)**, akkor a játék véget ér. Ha az aktuálisan elhelyezendő tetrisz elem kilógna a játéktérről jobbra (csak akkor fordulhat elő szabályosan, ha az utolsó oszlopot választotta a felhasználó), akkor ezt jelezzük**(-1)**, de a játék folytatódhat új oszlop index megadásával. Ha az elem felfelé lógna ki a játéktérről(**-2)**, akkor is vége van a játéknak. Ha sikerült maradéktalanul elhelyezni az elemet (azaz minden polyomino szegmens a játéktérben van), akkor a nulla **(0)** értéket adjuk vissza.

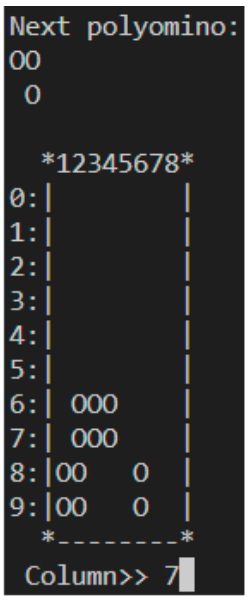
A tetrisz elem lehetséges landolási helyének kiszámítása (15 pont)
A polyominok legegyszerűbb felhelyezése a következő képpen történhet: a játékos által választott illetve a tőle jobbra található oszlopban (utóbbi csak, ha releváns: 1es elem mintánál nem fontos) meg kell keresnünk azt a szintet ahova a tetrisz elem elhelyezhető, de eggyel lejjebb már nem. (Mivel elértük a pálya alját vagy egy 'O' szegmens leendő helyén már van egy másik 'O'.)

Példa:

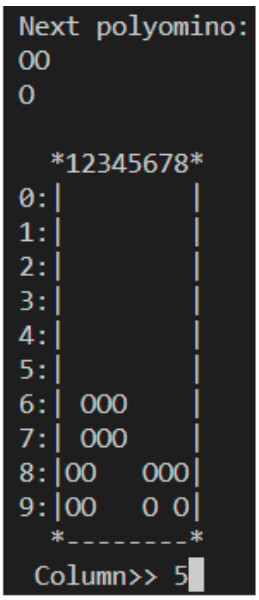
Az 1. ábrán az aktuális állás látható. A következő elem egy 4es (jobb felső háromszög), a játékosra várunk. A 3as oszlopot fogja megadni. A 3as és 4es oszlopban az első szabad sor a 6os. Azonban a polyomino alakját figyelembe véve egy szinttel lejjebb kell elhelyezni, így a landolási szint a 7es lesz



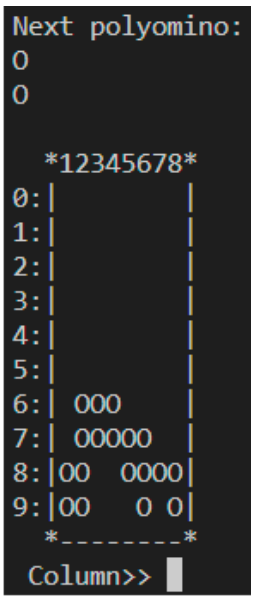
1. ábra



2. ábra



3. ábra



4. ábra

A 2. ábrán látható az elhelyezett elem. Itt a következő elem megint egy 4es lesz, melyet a 7es oszlopba ejt a játékos. A 7es és 8as oszlop üres tehát egészen a játéktér aljáig mehet az elem.

A 3. ábrán látható az eredmény. Ezután egy 3as (bal felső háromszög) elemet ejt a játékos az 5ös oszlopba.

A 4. ábrán látható, hogy az elem a 8as sorra fog illeszkedni, hisz „fennakad" a 6os oszlop korábbi elemein.

A tetrisz elem elhelyezése (4 pont)

Az elhelyezés során frissítsük a játéktér mátrixának megfelelő koordinátáit.

(Tipp: A játéktér kirajzolásakor az elhelyezett polyominot szemmel ellenőrizzük! Ha a tetrisz elem eredeti formájában, minden 'O' szegmense szabad mezőre esik valamint nem esett át korábbi elemeken, akkor sikeresnek tekinthetjük az elhelyezést.)

A függvény megkapja a játékteret és az aktuális polyominot tároló mátrixot valamint a játékos által választott oszlop indexet paraméterben

Főprogram – main() (10 pont)

A program összefogása és kommunikáció a felhasználóval. Köszöntsük a játékost és röviden ismertessük a játékot. (Saját szavakkal, pár mondatban elegendő.) Deklaráljuk a játékhoz szükséges változókat, tárolókat. Hívjuk meg a játék inicializálását végző alprogramokat, majd írassuk ki a kezdő állapotot. A játékos a **'1-8'** karakterekkel vezérelheti a tetrisz elemek ledobását. Egy karakter beütése után az **<ENTER>** (newline karakter) beütésével válasszuk ki az oszlopot. Magát a newline karaktert ignoráljuk. A játékból az **EOF** megadásával léphetünk ki. Az érvénytelen karaktereket vessük el, ne akadályozzák a játék futását. Érvényes oszlop index esetén frissítsük a játékteret. Az update_game() visszaadatt értéke alapján vezéreljük a játékot. Sikeres elhelyezés esetén**(0)** generáljunk új polyomino-t, frissítsük a lerakott elemek számát és rajzoljuk ki az új állapotot. Jobbra kilógás esetén**(-1)** figyelmeztessük a játékost és kérjünk be új oszlop indexet az aktuális elemnek. Sikertelen lerakás esetén**(-2)**, vagyis nem fért be az elem a megadott oszlopba, a játék leáll. A játék végén írjuk ki a standard kimenetre a játékos pontjait, azaz mennyi tetrisz elemet tudott sikeresen lerakni.