ELJÁRÁSOK ÉS FÜGGVÉNYEK

GYAKORLÓ FELADATOK C#-BAN

TARTALOM

ELJÁRÁSOK

MEGOSZTOTT VÁLTOZÓK

Az Eljárások fejezetben lévő feladatoknak külön projektfájlokat hozzunk létre.

1. ISMÉTLÉS NÉLKÜLI MEGOSZTOTT TÖMB

Készítsünk olyan eljárást, amely egy megosztott tömb elemeit kéri be billentyűzetről (egész számok). Az elemek száma 5, ismétlődést nem engedünk meg (két egyenlő értékű elemet ne fogadjon el a program)! A sikeres adatfelvitelt követően írassuk ki a képernyőre a tömb elemeit. A projekt neve: IsmNelkuliMegosztottTomb.

Az elkészítendő eljárások:

- Kiirasok
- Adatbekeres
- Tombkiiras

2. MEGOSZTOTT TÖMB MAXIMUMA

Készítsünk olyan eljárást, amely egy megosztott tömb elemei közül meghatározza a legnagyobb elem értékét, és ezen értéket berakja egy megosztott **max** nevű változóba, majd ezt kiírja! A tömb 10 elemű legyen, és a programkódban tároljuk az értékeit. A projekt neve: MegosztottTombMaximuma.

Az elkészítendő eljárások:

- Programfej
- Maximumkereses
- Eredmenykiiras

FÜGGVÉNYEK

3. ÖSSZEGZÉS

Készítsünk függvényt, mely kiszámítja egy tömb elemeinek összegét. A függvény neve legyen: Osszegzes.

4. MAXIMUMKERESÉS

Az előbbi feladathoz készítsünk egy függvényt, mely visszaadja a megosztott tömb legnagyobb értékét. A függvény neve legyen: Maximumkereses.

5. ÉRTÉKKERESÉS

Írjunk függvényt, mely az előbbi feladatokban létrehozott megosztott tömbben megkeres egy adott értéket, és igaz/hamis eredményt ad vissza attól függően, hogy az szerepel-e benne, vagy sem.

6. TÖMB FELTÖLTÉSE

Írjunk függvényt, mely segítségével fel tudunk tölteni a billentyűzetről egy 5 elemű, egészekből álló megosztott tömböt.

7. OSZTHATÓSÁG

Készítsünk olyan függvényt, amely megadja, hogy egy megosztott tömbben hány 3-mal osztható, de 5-tel nem osztható szám van!

PARAMÉTERES ELJÁRÁSOK ÉS FÜGGVÉNYEK

8. MELYIK A NAGYOBB? - ELJÁRÁS

Készítsünk olyan paraméteres eljárást, amely paraméterként kap két (valós) számot, és kiírja a két szám közül a nagyobbik értékét! Amennyiben a két szám egyenlő, úgy az első szám értékét kell kiírni! Az eljárást a Main metódusból konkrét számokkal hívjuk meg.

9. MELYIK A NAGYOBB? – ELJÁRÁS 2.

Az előbbi feladatban elkészített paraméteres eljárást hívjuk meg úgy, hogy a billentyűzetről begépelt két számon fusson le!

10. MELYIK A NAGYOBB? - FÜGGVÉNY

A korábban készített hasonló feladat eljárását írjuk át úgy, hogy olyan függvény legyen belőle, amely paraméterként kap két számot, és visszaadja a két szám közül a nagyobbik értékét! Amennyiben a két szám egyenlő, úgy az első szám értékét kell visszaadni!

11. DUPLÁZÓ - FÜGGVÉNY

Készítsünk függvényt, amely egy valós számot kap paraméterként, majd visszaadja annak dupláját.

12. HATVÁNY - FÜGGVÉNY

Írjunk hatványfüggvényt, mely visszaadja a hatvány értékét. A hatványalap és a -kitevő legyen a függvény bemenő paramétere. A kitevő pozitív szám lehet, max. 255-ig.

13. KARAKTERTÉGLALAP - ELJÁRÁS

Írjunk egy paraméteres eljárást, mely a paraméterként megadott sor- és oszlopszámú karaktert (max. 255 db) ír ki a monitorra. A kiírandó karaktert is paraméterként kell megadni.

14. LEGHOSSZABB EGYENLŐ - ELJÁRÁS

Készítsünk olyan paraméteres eljárást, amely kiírja egy paraméterben megadott, int típusú tömbben a leghosszabb egyenlő elemekből álló szakasz hosszát!

15. LEGHOSSZABB EGYENLŐ - FÜGGVÉNY

Az előbbi eljárást írjuk át függvény formájában! A függvény adja vissza a leghosszabb egyenlő szakasz hosszát! Ezt írassuk ki a főprogramból!

16. VÉLETLEN TÖMB ISMÉTLÉS NÉLKÜL - FÜGGVÉNYEK

Készítsünk programot, amely egy tömböt feltölt véletlen elemekkel egy megadott intervallum elemei közül úgy, hogy két egyenlő érték ne forduljon elő a tömbben! Az intervallum kezdő és végértékeit paraméterként adjuk át. A programban elkészítendő függvények:

- **Létezik_eFv:** Visszaadja az igaz/hamis értéket annak megfelelően, hogy a paraméterként fogadott egész szám szerepel-e a szintén paraméterként fogadott egész tömbben.
- **VeletlenTombIsmetlesNelkulFv:** Visszaadja azt az egész tömböt, melyet feltöltött a paraméterben megadott két egész szám közötti véletlen értékekkel. A véletlenek között nem lehet két egyforma szám.

A függvény belsejéből kell meghívni a Letezik_eFv nevű függvényt, ami segít az ismétlődések kiküszöbölésében.

- **TombKiirasElj:** A paraméterként kapott egész típusú tömb elemeit kiírja egymás után egy sorban, vesszővel elválasztva.

ALGORITMUSOK KÓDOLÁSA

17. SHELL RENDEZÉS - KÓDOLÁS

A következő algoritmus a Shell rendezés algoritmusával rendezi az N elemű (N<100) vektorban megadott számokat növekvő sorrendben. Kódolja az algoritmust a választott programozási nyelven! Az elkészült program forráskódját mentse **shellsort** néven!

A megoldás során vegye figyelembe a következőket:

- A választott programozási nyelvtől függően eltérő jelölésű operátorokat, adattípusokat és függvényeket kell alkalmaznia.
- A ”div” az egészosztás operátora.
- Az egész típusú változókhoz és vektorokhoz használjon 32 bites előjeles adattípust!

```
Eljárás ShellRendezes(a:Tömb[0..N] Egész)
   Változó gap, n, i, j, x, y : Egész
   gap := 1
   n := a.Hossz //a vektor elemszáma
   Ciklus amíg (gap * 2 <= n)
      gap := gap * 2
   Ciklus vége
   gap := gap - 1
   Ciklus
      i := 0
      Ciklus amíg ((i <= gap) ÉS (i + gap < n))
         j := i + gap
         Ciklus amíg (j < n)
            x := a[j]
            y := j - gap
            Ciklus amíg ((y > -1) ÉS (x < a[y]))
               a[y + gap] := a[y]
               y := y - gap
            Ciklus vége
            a[y + gap] := x
            j := j + gap
         Ciklus vége
         i := i + 1
      Ciklus vége
      gap := gap div 2
   amíg (gap > 0)
   Ciklus vége
Eljárás vége

Program shellsort
   Változó t: Tömb[0..9] Egész
   t[0] := 63
   t[1] := 54
   t[2] := 33
   t[3] := 45
   t[4] := 23
   t[5] := 99
   t[6] := 43
   t[7] := 10
   t[8] := 35
   t[9] := 87
   ShellRendezes(t)
   Ciklus i:=0 -tól 9 -ig (+1 lépésközzel)
      Ki: t[i]
   Ciklus vége
Program vége.
```

/Informatikai alapismeretek — emelt szintű érettségi (2016. okt.)/

18. SZÁMJEGYEK - KÓDOLÁS

A következő algoritmus egy N jegyű (N>=2) szám összes számjegyének egyszeri felhasználásával készíthető számok közül meghatározza az eredeti számnál nagyobb, legkisebb számot. Ha nem létezik ilyen szám, akkor az eredeti számot írja ki.

A szám jegyeit az alábbi karakter típusú vektorban tárolja és kezeli:

Változó: szam[0..N] Karakter. Kódolja az algoritmust a választott programozási nyelven! Az elkészült program forráskódját mentse NLSz néven!

A megoldás során vegye figyelembe a következőket:

- A ”szam” változó (paraméter) típusát a feladatkiírás tartalmazza.
- A választott programozási nyelvtől függően eltérő jelölésű operátorokat, adattípusokat és függvényeket kell alkalmaznia.
- A ”Térj vissza” utasítás megszakítja a függvény futását és meghatározza annak visszatérési értékét.

```
Konstans N=5
Függvény Cserel(szam, i1:Egész, i2:Egész):Tömb[0..N] Karakter
   Változó ch:Karakter
   ch:=szam[i1]
   szam[i1]:=szam[i2]
   szam[i2]:=ch
   Cserel:=szam
Függvény vége

Függvény Keres(szam): Tömb[0..N] Karakter
   Változó indA, indB, i, j, meddig:Egész
   indA:=-1
   i:=N-1
   Ciklus amíg i>0 és indA = -1
      Ha szam[i-1]<szam[i] akkor
         indA:=i-1
      Elágazás vége
      i:=i-1
   Ciklus vége
   Ha indA=-1 akkor
      Térj vissza szam
   Elágazás vége
   indB:=indA+1
   Ciklus j:=indA+2 -től N-1 -ig (+1 lépésközzel)
      Ha szam[j]>szam[indA] és szam[j]<szam[indB] akkor
         indB:=j
      Elágazás vége
   Ciklus vége
   szam:=Cserel(szam,indA,indB)
   Ciklus meddig:=N-1 -től 1 -ig (-1 lépésközzel)
      Ciklus j:=indA+1 -től meddig-1 -ig (+1 lépésközzel)
         Ha (szam[j]>szam[j+1] akkor
            Cserel(szam,j,j+1)
         Elágazás vége
      Ciklus vége
   Ciklus vége
   Keres:=szam
Függvény vége

Program NLSz:
   Változó szam /*típusa a feladatkiírásban*/
   szam[0]=’5’
   szam[1]=’2’
   szam[2]=’6’
   szam[3]=’3’
   szam[4]=’1’
   Ki: szam
   Ki: Keres(szam)
Program vége.
```

/Informatikai alapismeretek — emelt szintű érettségi (2016. máj.)/