emter

INFORMATIKA – javítókulcs

2016

ELMÉLETI TÉTEL:

„Járd körbe” a tömb fogalmát (Pascal vagy C/C++): definíció, egy-, két-, több-dimenziós tömbök, kezdőértékadás definíciókor, tömb típusú paraméterek átadása alprogramoknak. [2 pont]

GYAKORLATI TÉTEL:

1. Legyen az alábbi pszeudokód program (a beolvasott érték természetes szám; a „/” operátor osztási hányadost, a „%”operátor pedig osztási maradékot ad meg):

```
beolvas a
┌amíg a > 0 végezd
│ kiír a % 10
│ a = a / 10
└■
```

Mit ír ki a program, ha a beolvasott érték 2016? [0.5 pont] 6102

2. Legyen az alábbi pszeudokód program (a beolvasott érték természetes szám; a „/” operátor osztási hányadost, a „%”operátor pedig osztási maradékot ad meg):

```
beolvas a
k1 = 0
k2 = 0
┌amíg a > 0 végezd
│ ┌ha a % 2 == 1 akkor
│ │ k1 = k1 * 10 + a % 10
│ │különben
│ │ k2 = k2 * 10 + a % 10
│ └■
│ a = a / 10
└■
kiír k1, ”,”, k2
```

Mit ír ki a program, ha a beolvasott érték 2016? [0.5 pont] 1,602

3. Legyen az alábbi pszeudokód programrészlet (az x[1..n] (n≥1) tömbszakasz egy növekvő sorrendbe rendezett természetesszám- sorozatot tárol, az n változó pedig ennek hosszát):

```
beolvas a
i = n
┌amíg i > 0 ÉS x[i] ≥ a végezd
│ x[i+1] = x[i]
│ i = i - 1
└■
x[i+1] = a
┌minden i = 1,n+1 végezd
│ kiír x[i]
└■
```

Mit ír ki a program, ha n=5, a számsorozat elemei 13, 17, 23, 23, 53, a beolvasott érték pedig 55? [0.5 pont] 13, 17, 23, 23, 53,55

Mit valósít meg a programrészlet (fogalmazd meg tömören)? [0.5 pont] Beszúrja a-t, a helyére, a növekvő sorozatba

4. Legyen az alábbi pszeudokód programrészlet (a beolvasott értékek természetes számok; a „/” operátor osztási hányadost, a „%”operátor pedig osztási maradékot ad meg):

```
beolvas n
┌minden i = 1,n végezd
│ beolvas x
│ ┌amíg x > 9 végezd
│ │ s = 0
│ │ ┌amíg x > 0 végezd
│ │ │ s = s + x%10
│ │ │ x = x / 10
│ │ └■
│ │ x = s
│ └■
│ kiír x
└■
```

Mit ír ki a program, ha n-be az 5 értéket, majd az x változóba sorra az 139, 999, 89, 6, 11 értékeket olvassuk be? [0.5 pont] 4,9,8,6,2

5. Legyen az alábbi pszeudokód programrészlet (az y[1..n][1..n] (n≥1) tömb egy négyzetes mátrixot tárol, az n változó pedig ennek méretét; figyelem az aláhúzott változók sorrendjére):

```
i = n
┌amíg i > 0 végezd
│  j = n
│ ┌ amíg j > 0 végezd
│ │  kiír y[j,i]
│ │  j = j - 1
│ └■
│  i = i - 1
└■
```

Mit ír ki a program, ha n=4, a mátrix elemei pedig (sorfolytonosan) 1,2,3,4, 9,9,9,9, 1,2,3,4, 1,2,3,4? [0.5 pont] 4,4,9,4,3,3,9,3,2,2,9,2,1,1,9,1

6. Írj Pascal vagy C/C++ programot, amely billentyűzetről beolvassa az n értéket (1≤n≤999), valamint egy n elemű számsorozatot (elemei egész számok), majd kiírja a képernyőre a számsorozat elemeinek összegét. [1 pont]

```c
#include <stdio.h>
int main(){
	int n, szam, i, s = 0;
	scanf("%i", &n);
	for( i = 0 ; i < n ; ++i ){
		scanf("%i", &szam);
		s += szam;
	}
	printf("Osszeg: %i\n", s);
	return 0;
}
```

7. Írj Pascal vagy C/C++ programot, amely a matrix.txt állományból beolvassa az n és m értékeket (2≤n,m≤999), valamint egy nxm méretű mátrixot (elemei egész számok), és kiírja a képernyőre, hogy a mátrix hány sora alkot szigorúan növekvő számsorozatot. [1 pont]

```
#include <stdio.h>
int main(){
	int n, m, x, y, i, j, db = 0;//x,y adott sor ket egymasutani elemet tarolja
	bool kem;
	freopen("matrix.txt", "r", stdin); //az allomanyt standard input-ta teszuk
	scanf("%i%i", &n, &m);
	for( i = 0 ; i < n ; ++i ){
		scanf("%i", &x); // beolvassuk a sorelsot
		kem = true;
		for( j = 1 ; j < m ; ++j ){
			scanf("%i", &y);
			if( x >= y ){
				kem = false;
			}
			x = y;
		}
		if(kem){
			++db;
		}
	}
	freopen("CON", "r", stdin);//ujra a billentyuzet legyen a standard input
	printf("Ennyi sora szigoruan novekvo: %i\n", db);
	return 0;
}
```

8. Írj függvényt (Pascal vagy C/C++ nyelven), amely visszatéríti egy keresett érték pozícióját egy növekvő sorrendbe rendezett számsorozatban. Tekintsd úgy, hogy a keresett értéket az `x` változó, a számsorozat hosszát az `n` változó, a számsorozatot pedig az `a[1..n]` tömb (Pascal változat) vagy az `a[0..n-1]` tömb (C/C++ változat) tárolja. Ha többször is előfordul az illető érték a számsorozatban (monoton növekvő), akkor az utolsó előfordulási pozícióban vagyunk érdekeltek. Ha nem szerepel a keresett érték a számsorozatban, akkor a függvény térítse vissza a `-1` értéket. *A függvényt implementáld iteratívan is és rekurzívan is.* Mindkét esetben add meg azt is, hogy miként hívnád meg függvényedet. Törekedj hatékony megoldásra! [1 pont]

```
int rekurziv(int *a, int kezdet, int veg, int x){ //meghivas: rekurziv(a,0,n-1,x)
	if( kezdet > veg ){ return -1; }
	int kozep = (kezdet + veg) / 2;
	if( x == a[kozep] ){
		while( kozep < veg && x == a[kozep+1] ){
			++kozep;
		}
		return kozep;
	}
	else if( x < a[kozep] ){
		return rekurziv(a,kezdet,kozep-1,x);
	}
	else{
		return rekurziv(a,kozep+1,veg,x);
	}
}
int iterativ(int *a, int n, int x){ //meghivas: iterativ(a,n,x)
	int kezdet = 0, veg = n-1, kozep;
	while( kezdet <= veg ){
		kozep = (kezdet + veg) / 2;
		if( x == a[kozep] ){
			while( kozep < veg && x == a[kozep+1] ){
				++kozep;
			}
			return kozep;
		}
		else if( x < a[kozep] ){
			veg = kozep-1;
		}
		else{
			kezdet = kozep+1;
		}
	}
	return -1;
}
```

9. Egy sajátos módszert definiálunk az {1,2,3,…,n} halmaz valamely permutációjának rendezésére, amelyben egy *lépés* abból áll, hogy a számsorozat valamely elemét a számsorozat végére tesszük. Írj függvényt (Pascal vagy C/C++ nyelven), amely visszatéríti a minimális *lépés*-számot, amellyel a paraméterként kapott permutáció (mint `n` elemű számsorozat) elemi növekvő sorrendbe rendezhetőek. Például a 3,1,5,2,4 permutáció/számsorozat esetén a minimális lépésszám 3. (Példa egy *tetszőleges* lépésre: az adott példában az 5-ös elemet a számsor végére tesszük; e lépés nyomán a számsor 3,1,2,4,5 –re változik) [1 pont]

Megjegyzés: Megszámoljuk az 1-es értékkel kezdődő leghosszabb, egymás utáni értékeket tartalmazó, nem föltétlenül egymás utáni pozíciókban levő részsorozat hosszát. A minimális lépésszámot úgy kapjuk, hogy n-ből kivonjuk a fentebb meghatározott értéket. A példára, azon részsorozat, amelynek elemeit nem kell végére-léptetni:1,2 (3,1,5,2,4). A többi elemet (5-2=3 darabot), növekvő sorrendben, egyenként végére-léptetjük: (**3**,1,5,2,4) >> (1,5,2,**4**,3) >> (1,**5**,2,3,4) >> (1,2,3,4,5)

```
int minlepes(int *a, int n){
      int i, p;
      p = 1;
      for( i = 0 ; i < n ; ++i ){
            if( a[i] == p ){
                  ++p;
            }
      }
      return n - (p - 1);
}
```