emter

INFORMATIKA – JAVÍTÓKULCS

2015. március 28.

Munkaidő: 3 óra
Megjelenésre: 1 pont

ELMÉLETI TÉTEL (összesen 2 pont)

ALPROGRAMOK (térj ki olyan fogalmakra, mint: formális és aktuális/effektív paraméterek; érték/cím-szerinti paraméterátadás, …, adj példákat, stb)

GYAKORLATI TÉTEL (összesen 7 pont)

[0.5 pont] 1. Mit ír ki az alábbi pszeudokód program, ha a beolvasott érték (természetes szám) 20150328 (a „/” operátor osztási hányadost ad meg)? Mit valósít meg a program (fogalmazd meg tömören)?

```
beolvas n;
┌amíg n > 9 végezd
│     n = n / 10;
└■
kiír n;
```

Helyes válasz: 2; A program kiírja a beolvasott természetes szám első számjegyét.

[0.5 pont] 2. Mit ír ki az alábbi pszeudokód program, ha a beolvasott érték rendre 0, 1, 2, …, 10? Mit valósít meg a program (fogalmazd meg tömören)?

```
beolvas n;
x11=1; x12=1; x21=1; x22=0;
y11=1; y12=1; y21=1; y22=0;
┌minden i=1,n végezd
│     z11=x11*y11+x12*y21;
│     z12=x11*y12+x12*y22;
│     z21=x21*y11+x22*y21;
│     z22=x21*y12+x22*y22;
│     x11=z11; x12=z12; x21=z21; x22=z22;
└■
kiír x11;
```

Helyes válasz: 1,2,3,5,8,13,21,34,55,89,144; A program kiírja az (n+2)-dik fibonacci számot (amennyiben $F_0=0$, $F_1=1$, $F_2=1$, …).

[0.5 pont] 3. Mit ír ki az alábbi pszeudokód program, ha a beolvasott érték 4?

```
beolvas n;
┌minden i=1,n végezd
│     ┌minden j=1,i végezd
│     │     x[j][i] = i*i - j + 1;
│     │     x[i][j] = (i-1)*(i-1) + j;
│     └■
└■
┌minden i=1,n végezd
│     kiír x[i][i];
└■
```

```
┌minden i=1,n végezd
│     kiír x[i][n-i+1];
└■
```

Helyes válasz: 1, 3, 7, 13; 16, 8, 6, 10.

[0.5 pont] 4. Mit ír ki az alábbi pszeudokód program, ha a beolvasott értékek 10, 2, 5, 7, 3, 9, 8, 0, 4, 1, 6. (a *min/max* függvény a kisebbik/nagyobbik paraméter értékét téríti vissza)

```
beolvas n;
┌minden i=1,n végezd
│     beolvas a[i];
└■
c[0] = 0;
c[1] = a[1];
┌minden i=2,n végezd
│     c[i] = min(c[i-1] + a[i] , c[i-2] + max(a[i] , a[i-1]));
└■
kiír c[n];
```

Helyes válasz: 31

[1 pont] 5. Milyen értéket térít vissza az alábbi C/C++ vagy Pascal függvény, ha a paraméter értéke 34571, illetve ha 234950? (a *min* függvény a kisebbik paraméter értékét téríti vissza)

```
int R(int n)
{
      if (n < 10) return n;
      return R(n/100)*10 + min((n%100)/10 , n%10);
}

function R(n: integer) : integer;
begin
      if n < 10 then
            R := n
      else
            R := R(n div 100)+ min((n mod 100) div 10 , n mod 10);
end;
```

Helyes válasz (C/C++): 341; 240.
Helyes válasz (Pascal): 8; 6.

[0.5 pont] 6. Írj Pascal vagy C/C++ programot, amely beolvas a billentyűzetről egy *n* ($1 \le n \le 999$) értéket és egy *n* elemű egész számokat ($-10^9..10^9$ tartományból) tartalmazó számsorozatot, és kiírja a képernyőre azt, hogy a számsorozat hány elemére igaz, hogy egyenlő a szomszédjai számtani közepével.

Helyes válasz (C/C++):

```
int main(){
      int n, i, bal, kozep, jobb, k;
      scanf("%i", &n);
      if (n<3){
            printf("darab: 0\n");
            return 0;
      }
      scanf("%i%i", &bal, &kozep);
      k = 0;
      for(i=3 ; i<= n ; ++i){
            scanf("%i", &jobb);
            if ((double)kozep == ((bal+jobb)/2.0)){
                  ++k;
            }
            bal = kozep;
            kozep = jobb;
      }
      printf("darab: %i\n", k);
      return 0;
```

```
}
```

[0.5 pont] 7. Írj Pascal vagy C/C++ programot, amely állományból beolvassa az *n* (1≤*n*≤10) és *m* (1≤*m*≤10) értékeket és egy *nxm* méretű egész számokat (-999..999 tartományból) tartalmazó mátrixot, és kiírja a képernyőre a mátrixot, táblázatos alakban úgy, hogy minden páratlan sorszámú sor fordított sorrendben jelenjen meg.

Példa:

```
Bemenet          Kimenet
3 4                 4   3   2   1
1 2 3 4            11   2  33   4
11 2 33 4         444   3   2 111
111 2 3 444
```

Helyes válasz (C/C++):

```
int main(){
	FILE *f;
	int n, m, i, j, *t, x;
	f = fopen("be7.txt", "r");
	if (f == NULL){
		printf("Sikertelen allomany megnyitasi kiserlet!");
		return 0;
	}
	fscanf(f, "%i%i", &n, &m);
	t = (int*)malloc(m*sizeof(int));
	for(int i=0 ; i<n ; ++i){
		if(i%2){
			for(int j=0 ; j<m ; ++j){
				fscanf(f, "%i", &x);
				printf("%4i", x);
			}
		}
		else{
			for(int j=0 ; j<m ; ++j){
				fscanf(f, "%i", &t[j]);
			}
			for(int j=m-1 ; j>=0 ; --j){
				printf("%4i", t[j]);
			}
		}
		printf("\n");
	}
	return 0;
}
```

[1.5 pont] 8. Adott *n* processzor és *m* elvégzendő feladat (ismertek a feladatok elvégzéséhez szükséges t_1, t_2, …t_m időegységek). Osszuk ki úgy a feladatokat a processzoroknak, hogy az összvégrehajtási idő minimális legyen. Írj Pascal vagy C/C++ programot, amely beolvassa állományból az *n* (1≤*n*≤999) és *m* (1≤*m*≤999999) értékeket és a t_1, t_2, …t_m számsorozatot (egész számok az 1..999 tartományból), majd kiír a képernyőre egy számot, a kapott minimumértéket.

Helyes válasz (C/C++):

(Bár nem garantál optimumot, az alábbi mohó megoldás, vagy ezzel egyenértékű, maximális pontot ért)

```
int intcmp(const void *p1, const void *p2){
	int *q1 = (int*)p1;
	int *q2 = (int*)p2;
	return *q2 - *q1;
}
int main(){
	int n, m, *t;
	long p[1000]={0};
	FILE *f;
	f = fopen("be8.txt", "r");
	if (f == NULL){
```

```
            printf("Sikertelen allomany megnyitasi kiserlet!");
            return 0;
      }
      t = (int*)malloc(1000000*sizeof(int));
      if (t == NULL){
            printf("Sikertelen helyfoglalasi kiserlet!");
            return 0;
      }
      fscanf(f, "%d%d", &n, &m);
      for(int i=0 ; i<m ; ++i){
            fscanf(f, "%d", &t[i]);
      }
      fclose(f);
      //csökkenő sorrendbe rendezzük a feladat-időket
      qsort(t,m,sizeof(int),intcmp);
      //a p tömb növekvő sorrendben tárolja a processzorok leterheltségi fokát
      for(int i=0 ; i<m ; ++i){
            p[0] += t[i]; // p[0] - legkevésbé leterhelt processzor
            long x = p[0];
            int k;
            for (k = 1 ; k<n ; ++k){ //helyreállítjuk a növekvő sorrendet
                  if ( (p[k-1] = p[k]) >= x){
                        break;
                  }
            }
            p[k-1] = x;
      }
      printf("Minimalis osszido: %ld\n", p[n-1]);
      return 0;
}
```

[1.5 pont] 9. Írj Pascal vagy C/C++ programot, amely beolvassa a billentyűzetről az *n* ($1 \le n \le 10$) és *p* ($1 \le p \le 10$) értékeket, és generálja egy kimeneti állományba az {1, 2, ..., *n*} halmaz *p*-enkénti kombinációit.

Helyes válasz (C/C++):

```
#define N 11
void kiiras(int k, int *x){
      printf("{");
      for(int i=1 ; i< k ; ++i){
            printf("%i,", x[i]);
      }
      printf("%i}\n", x[k]);
}
void bt(int k, int *x, int n, int p){
      if (k==p){
            kiiras(k,x);
      }
      else{
            for(x[k+1]=x[k]+1 ; x[k+1]<= n-p+k+1 ; ++x[k+1]){
                  bt(k+1,x,n,p);
            }
      }
}
int main(){
      int k, n, p, x[N];
      scanf("%i%i", &n, &p);
      x[0] = 0;
      freopen("ki9.txt","w",stdout);
      bt(0,x,n,p);
      return 0;
}
```

Megjegyzés: *Törekedjetek hatékony programokat írni (6-9 feladatok), úgy a tárhelyigény, mint a futási idő szempontjából!*