BABEŞ-BOLYAI UNIVERSITY
FACULTY OF MATHEMATICS AND COMPUTER SCIENCE

Math-CompSci Contest - model

Written Test for Computer Science

Subject A (30 points)

1. **(5p)** Consider the following subalgorithm:

```
Subalgorithm f(a):
       If a != 0 then
              return a + f(a - 1)
       else
              return 0
       EndIf
EndSubalgorithm
```

Which of the following statements is false?

a. *f* is a recursively defined subalgorithm
b. if ***a*** is negative, the subalgorithm returns 0
c. the value calculated by ***f*** is ***a*** * (***a*** + 1) / 4
d. the subalgorithm computes the sum of the natural numbers smaller or equal to ***a***
e. calling ***f***(-5) results in an infinite cycle.

2. **(5p)** Consider the following subalgorithm:

```
Subalgorithm f(a, b):
       If (a > 1) then
              return b * f(a - 1, b)
       else
              return 1
       EndIf
EndSubalgorithm
```

How many times is function ***f*** called in the following code snippet:

```
x ← 4;
y ← 3;
z ← f(x, y);
```

a. 4 times
b. 3 times
c. infinite number of times
d. never
e. once

3. **(5p)** Consider all arrays having length $\boldsymbol{l} \in \{1, 2\}$ built using letters from the set {*a, b, c, d, e*}. How many of them are sorted strictly increasing and have an even number of vowels? (*a* and *e* are vowels)

a. 7
b. 80
c. 81
d. 78
e. 2

4. **(5p)** A matrix has 8 rows, contains only 1's and 0's and has the following three properties:
   a. there is a single element with value 1 on the first row,
   b. row $\boldsymbol{j}$ contains twice as many non-zero elements as row $\boldsymbol{j} - 1$, for all $\boldsymbol{j} \in \{2, 3, \ldots, 8\}$,
   c. on the last row there is a single element with value 0.

   What is the total number of 0 elements in the matrix?

a. 777
b. 769
c. 528
d. there is no such matrix
e. 1

5. **(5p)** Consider 3 arrays named ***a***, ***b***, ***c*** having ***n***, ***m***, ***k*** elements respectively and the following subalgorithms:

```
Subalgorithm F1(x, l):
        s ← 0
        For i ← 1, l execute
                s ← s + x[i]
        EndFor
        return s
EndSubalgorithm
```

```
Subalgorithm F2(n1, n2):
        return n1 + n2
EndSubalgorithm
```

Which of the following instructions are correct in the case of arrays (***a***, ***b***, ***c***) containing ***n***, ***m***, ***k*** natural numbers, respectively:

```
a. F2(F2(F1(a,n), F1(b,m)), k)
b. val = F1(c,k) + F2(F1(b,m), F1(a,n))
c. val = F1(c,k) + F2(F1(a,m), b,n)
d. F2(F2(F1(a,n), F1(b,m)), F1(c, k))
e. val = F1(k, c) + F2(F1(m, b), F1(n, a))
```

6. **(5p)** Consider the following subalgorithm:

```
Subalgorithm fc(a, s):
        k = 0
        For i ← 1, length(s) execute:
                k = k + a
        EndFor
        return k
EndSubalgorithm
```

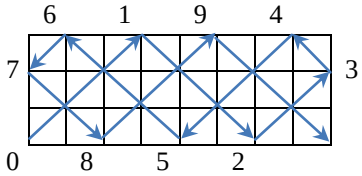
Which of the code snippets below must be executed to display 75?
Observation: arrays are indexed starting with 1.

a. nr = fc("ana", 25)
   display(nr)
b. nr = fc(25, "ana")
   display(nr)
c. display(fc(25, "ana"))
d. there is no such code snippet
e. display(fc("ana", 25))

<u>Subject B</u> (60 points)

1. Ray (25 points)

We have a rectangle bordered with mirrors facing inwards on all sides. A ray of light starts from the left bottom edge of the rectangle making a 45^o angle with the bottom side and hits one of the top or right sides of the rectangle. Here it is reflected, and starts towards another side, at the same 45^o angle with the side it reflected from. The ray continues to reflect from the sides of the rectangle until it reaches one of the corners.



Write a subalgorithm that calculates the number of times (***nrChange***) the ray changed direction before it stopped in a corner. The starting point is not counted. The input parameters are the width ($1 < a < 10\,000$) and height ($1 < b < 10\,000$) of the rectangle, and ***nrChange*** is the output parameter (***a***, ***b***, ***nrSchimb*** $\in \mathbf{N}$).
Example 1: if ***a*** = 8 and ***b*** = 3, then ***nrChange*** = 9.
Example 2: if ***a*** = 8 and ***b*** = 4, then ***nrChange*** = 1.

2. Viruses (15 points)

Within an experiment, a population of n ($3 \leq n \leq 1\,000$) viruses can evolve as follows:

a. if at the beginning of an hour, the population has an *even* number of viruses, at the end of the hour the population will decrease by 50%.

b. if at the beginning of an hour, the population has an *odd* number of viruses, at the end of the hour the population will increase by 1;

c. if at the end of an hour, the virus population is strictly lower than a *critical survivability threshold*, the population dissapears.

Write a subalgorithm that determines the number of hours (***nrHours***) required to destroy an initial population of n viruses, knowing the critical survivability threshold k ($2 \leq k < n$). Input parameters are n and k, and ***nrHours*** is the single output parameter.

Example: if $n = 11$ and $k = 3$, the population is destroyed in ***nrHours*** = 5.

3. Sorting (10 points)

Consider the following subalgorithm:

```
1:    Subalgorithm sort(a, n):
2:      If n > 0 then
3:         sort(a, n - 1)
4:         x ← a[n]
5:         j ← n - 1
6:         While (j >= 0 and a[j] > x) execute:
7:                j ← j - 1
8:         EndWhile
9:         a[j + 1] ← x
10:     EndIf
11:   EndSubalgorithm
```

What instruction/instructions must be added, and where, so that following the *sort*(***a***, ***n***) call array ***a*** containing ***n*** natural numbers will be sorted?

4. Control digit (10 points)

Consider the following subalgorithm that determines the control digit for a natural number consisting of at least 2 digits.

```
1:   Subalgorithm controlDigit(x):
2:       While x > 9 execute:
3:         s ← 0
4:         While x > 0 execute:
5:            s ← s + x MOD 10 { x mod 10 is the remainder of dividing x by 10}
6:            x ← x DIV 10    { x div 10 is the quotient of dividing x by 10}
7:         EndWhile
8:         x ← s
9:       EndWhile
10:      return x
11:  EndSubalgorithm
```

Replace this subalgorithm's body with maximum 2 instructions so that the resulting subalgorithm has the same effect.

Notă:

1. All subjects are mandatory
2. Solutions must be written on the exam sheets in detailed form (drafts are not graded).
3. Default 10 points.
4. Working time is 3 hours.

Evaluation

Granted points ..... **10 points**

Subject A ..... **30 points**

A. 1. Answers b, c, d ..... **5 points**

A. 2. Answer a ..... **5 points**

A. 3. Answer a ..... **5 points**

A. 4. Answer a ..... **5 points**

A. 5. Answer b ..... **5 points**

A. 6. Answers b, c ..... **5 points**

Subject B ..... **60 points**

B. 1. Ray ..... **25 points**

V1: correct determination of value ***nrChange*** by using *gcd*(***a***, ***b***) ..... 25 points

- *gcd*(***a***, ***b***) (or *scm*(***a***,***b***)) ..... 10 points
- calculation of ***nrChange*** ..... 15 points

V2: correct determination of value ***nrChange*** by using another algorithm (simulation) ..... 15 points

B. 2. Viruses ..... **15 points**

- iterativ or reccursive problem solving ..... 10 points
- computation (population dissapears at the end of an hour) ..... 5 points

B. 3. Sorting ..... **10 points**

- statement identification `(a[j + 1] ← a[j])` ..... 5 points
- insertion between rows 6 and 7 ..... 5 points

B. 4. Control digit ..... **10 points**

- control digits cand be computed as ***nr*** `mod 9` ..... 10 points

Solution

Solution – Subject B.1.: Ray ..... 25 points

```
//greatest common divisor of 2 numbers a and b
int cmmdc(int a, int b){
	if ((a == b) && (a == 0))
		return 1;
	if (a * b == 0)
		return a + b;
	while (a != b)
		if (a > b)
			a -= b;
		else
			b -= a;
	return a;
}

// computing nrChange
int raza(int a, int b){
	int d = cmmdc(a, b);
	return b / d + a / d - 2;
}
```

Solution – Subject B.2.: Viruses ..... 15 points

```
//computing the nrHours
int virusi(int n, int k){
	bool distrus = (n < k);
	int nrOre = 0;
	while (!distrus){
		if (n % 2 == 0)		//daca avem nr par de virsui, injumatatim populatia
			n = n / 2;
		else			//daca avem nr impar de virsui, marim populatia cu un virus
			n = n + 1;
		nrOre = nrOre + 1;
		distrus = (n < k);	//verificam daca populatia dispare
	}
	return nrOre;
}
```

Solution – Subject B.3.: Sorting ..... 10 points

```
 1:    Subalgorithm sorting(a, n):
 2:      If n > 0 then
 3:         f(a, n - 1)
 4:         x ← a[n]
 5:         j ← n - 1
 6:         While (j >= 0 and a[j] > x) do:
 7:                a[j + 1] ← a[j]
 8:                j ← j - 1
 9:         ENdWhile
10:         a[j + 1] ← x
11:      EndIf
12:    EndSubalgorithm
```

Solution - Subject B. 4. Control digit ..... 10 points

```
 1:   Subalgorithm controlDigit(x):
 2:       Return x mod 9
 3:   EndSubalgorithm
```