Süleyman Gölbol
76210123

The time of writing the exam is 62 minutes and 19 seconds. Good luck!

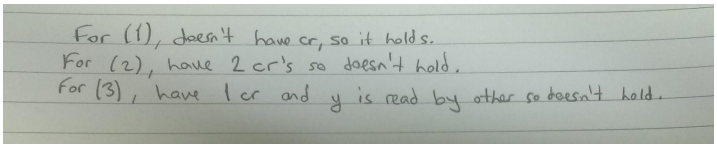
1. naloga: Take a look at this (flawed) distributed program:

```
MPI_Init(&argc, &argv);
MPI_Comm_size(comm, &comm_size);
MPI_Comm_rank(comm, &comm_rank);
dest = (comm_rank + 1) % comm_size;
source = (comm_rank + comm_size - 1) % comm_size;
MPI_Send(vector_A, vector_size, MPI_INT, dest, tag, comm);
MPI_Recv(vector_B, vector_size, MPI_INT, source, tag, comm, &status);
MPI_Finalize();
```

- Show why this program will not work properly!
- Fix the program so that it works properly (comment your changes)!

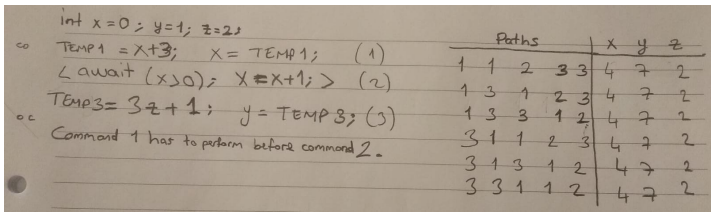
2. naloga: Specify the definition of one-time property and for each command in the program below, check if this property holds:

```
int x = 0, y = 10, z = 6
co while (x != y) x = x + 6; (1)
|| z = y - x;               (2)
|| y = z;                   (3)
oc
```



3. naloga:

```
int x = 0;y=1;z=2;
co x = x + 3;
|| <await x > 0; x = x + 1;>
|| y = 3 * z + 1;
oc
```



List all possible program traces and give the final value for each trace. Comment each solution!

4. naloga:

1. Write a program in the Java programming language where two threads access the common object of the *Print* class.
2. Class Print contains only one variable type String, whose value is assigned by the constructor.
3. Both threads access the object of the class Print and print in reverse order the value of the variable type String, letter by letter, where each letter is printed with a hyphen (–).

Example for *FAMNIT*: T–T–I–I–N–N–M–M–A–A–F–F

REVERSE PRINTING CODE

```
1  public class ReverseAccess{
2
3      public static void main(String[] argc){
4          String string = "FAMNIT";
5          Print objectToPrint = new Print(string);
6
7          Thread thread1 = new Thread( new Print(string) );
8          Thread thread2 = new Thread( new Print(string) );
9
10         thread1.start();
11         thread2.start();
12
13         try{
14             thread1.join();
15             thread2.join();
16         }catch(InterruptedException e){
17             e.printStackTrace();
18         }
19         System.out.println();
20     }
21
22 }
23
24 class Print implements Runnable{
25     private String value;
26
27     public Print(String value){
28         this.value = value;
29     }
30
31     @Override
32     public void run(){
33         for(int i=0; i<value.length(); i++){
34             if(i==value.length())
35                 System.out.print(value.charAt(value.length()-i-1) );
36             else
37                 System.out.print(value.charAt(value.length()-i-1) + "-");
38             Thread.yield();
39             try{
40                 Thread.sleep(10);
41             }catch(InterruptedException e){
42                 e.printStackTrace();
43             }
44         }
45     }
46 }
```