Beginselen van Programmeren
Exercise Session 4:
correctness proofs

1 Warm up exercises

E1 - What!? Why?? How...?

1. Explain why it is important to specify the precondition and postcondition of an algorithm.
2. What is an invariant? Why do you need it?
3. How do you find an invariant?
4. Why do you need to prove finiteness if you have already proven the steps related to the invariant (Steps 3.1-4)?
5. How does a variant help to prove finiteness?
6. How do you find a variant?

E2 - revisit from Exercise session 3. Write a program that computes the factorial of a positive integer, using a while loop. $n! = 1 * 2 * 3 * \cdots * n$
Proof the correctness of your solution.

2 Now let's continue...

E3. - Proof the correctness of the algorithm below that counts the number of capital letters in a string.

```
amt = 0
len_string = len(s)
counter = 0

while(counter < len_string):
    if s[counter].isupper():
        amt+=1
    counter += 1
```

E4 - revisit from Exercise session 3. A prime number is one that is not divisible by any number other than 1 and itself. Write a program that asks the user to input an integer and checks whether the given integer is a prime number.
Use a while loop in your solution.
Proof the correctness of your algorithm.
Could you have chosen a different pre-condition? How would that affect the rest of your correctness proof?

E5 - revisit from Exercise session 3. Write a program to input a number and reverse the order of digits. For example if the input to the program is 12345, the output should be 54321. (*Do not use string*)

Proof the correctness of your algorithm.

E6 - linear search. Write a program that checks whether a number is in a list of numbers. Use a while loop in your solution.

Proof the correctness of your algorithm.

E7 - max. Write a program that searches the maximum value in a list of numbers. Use a while loop in your solution.

Proof the correctness of your algorithm.

E8 - gcd. Write a program that calculates the greatest common divisor of 2 natural numbers. Use a while loop in your solution.

Proof the correctness of your algorithm.

E9 - lcm. Write a program that calculates the least common multiple of 2 natural numbers. Use a while loop in your solution.

Proof the correctness of your algorithm.