CSC236H, Winter 2016
Assignment 4
Due March 25th, 11:59 p.m.

- You may work in groups of no more than **two** students, and you should produce a single solution in a PDF file named `a4.pdf`, submitted to MarkUs. Submissions must be **typed**.
- Please refer to the course information sheet for the **late submission policy**.

1. Consider the following code.
   Precondition: $a \in \mathbb{R}$ and $b \in \mathbb{N}$.

   ```
   def rec_exp(a, b):
       if b == 0:
           return 1
       else if b mod 2 == 0:
           x = rec_exp(a, b/2)
           return x * x
       else:
           x = rec_exp(a, (b - 1)/2)
           return x * x * a
   ```

   State a postcondition for this algorithm (your postcondition must involve exponentiation). Then, prove that the algorithm is correct with respect to your specification.

2. Consider the following algorithm.
   Precondition: $a, b \in \mathbb{N}$, and $b > 0$.
   Postcondition: $a = b \cdot q + r$ and $q \geq 0$ and $0 \leq r < b$.

   ```
   def Div(a, b):
       q = 0
       r = a
       while r ≥ b:
           q = q + 1
           r = r - b
       return [q, r]
   ```

   (a) Give an appropriate loop invariant for the purpose of proving both partial correctness and termination for the above program with respect to its given specification.
   (Hint: your loop invariant should relate a, b, q, and r).

   (b) Give a formal proof of the partial correctness of *Div* with respect to the given specification.

   (c) Give a formal proof of termination of *Div*.

3. Recall that a point p in the plane can be given by a pair of numbers (a, b) where a is the x-coordinate and b is the y-coordinate. For any two such points p given by (p_1, p_2), and q given by (q_1, q_2), one can calculate the distance between p and q by various formulas. In the following exercise you may assume that the function $Distance(p, q)$ terminates and returns the distance between any two points p, q, each given as a tuple of numbers.

   Write a detailed proof of correctness for the following algorithm.

   Precondition: A is a list of tuples of integers, and $len(A) \geq 2$, and $1 \leq e < len(A)$.
   Postcondition: Returns a pair of points in $A[0 : e + 1]$ with minimal distance.

```
def Find_Closest_Pair_Rec(A, e):
    if e == 1:
        return (A[0],A[1])
    (p, q) = Find_Closest_Pair_Rec(A, e - 1)
    min = Distance(p, q)
    i = 0
    while i < e:
        if Distance(A[e], A[i]) < min:
            min = Distance(A[e], A[i])
            p = A[e]
            q = A[i]
        i = i + 1
    return (p, q)
```