Regulations on assignments

- The assignments are graded according to the correctness, preciseness and legibility of the solutions. All handwritten parts, including figures, should be clear and legible. This assignment is marked out of 20 possible marks.

- Please submit your solution in onQ before the due time. The submission must be in one of formats: .PDF, .JPG, .PNG, .DOCX.

- **The assignment must be based on individual work.** Copying solutions from other students is a violation of academic integrity. See the course onQ site for more information.

1. (5 marks) What should the pre-condition `P` be in each of the following ten correctness statements for the statement to be an instance of Hoare's axiom scheme? All variables are of type `int`.

    (a) `P { x = 1; } x == 1`

    (b) `P { x = 1; } x == 2`

    (c) `P { x = y + z; } 0 < x + y + z`

    (d) `P { x = y*z + 3; } x*x > y + 2`

    (e) `P { z = y + 1; } Exists(z = 0; z < 10) z + y == 50`

    (f) `P { x = x + y; } Exists(y = 0; y < 15) x*x == y + t`

    (g) `P { x = y + 1; } ForAll(z = 1; z < 100) x + 2*y > z + 2`

    (h) `P { z = x + y; } ForAll(y = 1; y < x) x + y + z < 100`

    (i) `P { x = y + z; } Exists(x = 0; x < 10) z*z + 2*x == 15`

    (j) `P { x = y + z } Exists(y = 0; y < 100)(x + y == 15 || z*x + y < 100 )`

2. Verify the validity of the following correctness statements **by adding all the intermediate assertions** (that is, give the proof tableau). All variables are of type `int`. Clearly state any mathematical facts and inference rules used.

    (a) (2.5 marks)

```
ASSERT( x >= 5 || ( x <= 0 && y == 2 ) )
z = x - y;
y = y + z;
x = y - z;
ASSERT( x == 2 || y > 3 )
```

(b) (2.5 marks)

```
ASSERT( z >= 0 )
if ( x < 2 ) { y = z+3; } //end-if
z = x + z;
ASSERT( y > x + 1 ||  z > 1 )
```

3. (5 marks) Select a <u>loop invariant</u> and give a <u>complete proof tableau</u> (that is, <u>add all the intermediate assertions</u>) for the following correctness statement. Also *make an argument for termination*. All variables are of type integer.

```
ASSERT(k >= 0)
i = k;
sum = k;
while( i > 0 )
{
  i = i - 1;
  sum  = sum + i;
}
ASSERT(sum == Σ_{j=0}^{k} j )
```

Above $(\sum_{j=0}^{k} \text{j})$ denotes the sum $0 + 1 + 2 + \ldots + k$.

4. (5 marks) Write and verify a program that computes the sum of the squares of the n smallest positive odd integers. For example, when n is 4, the sum should be 1 + 9 + 25 + 49 == 84. Below is given the specification.

Declarative interface:

```
const int n; /* the program will compute the sum of the squares
                of the n smallest positive odd integers*/
int sum;     /* the sum is stored in this variable */
```

Below are given the pre- and post-condition:

```
ASSERT( n >= 1 )
/* the program goes here */
ASSERT( sum == $\sum_{i=0}^{n-1}$ (2*i+1)*(2*i+1) )
```

The program should use a while-loop and standard arithmetic operations.

Select a loop invariant and give a complete proof tableau for the program (including all the intermediate assertions). Also make an argument for termination.