Suggested Solutions for Homework Assignment #4

We assume the binding powers of the logical connectives and the entailment symbol decrease in this order: $\neg$, $\{\forall, \exists\}$, $\{\wedge, \vee\}$, $\rightarrow$, $\leftrightarrow$, $\vdash$.

1. Prove that the following annotated program segments are correct:

   (a) (10 points)

   $\{\mathit{isGCD}(x, y, c)\}$
   S1: **if** B: $x < y$ **then** S2: $x, y := y, x$ **fi**;
   S3: $x := x - y$
   $\{\mathit{isGCD}(x, y, c)\}$

   The predicate *isGCD* is as defined in HW#2.

   Solution. We note that $\mathit{isGCD}(x, 0, |x|)$ and $\mathit{isGCD}(0, x, |x|)$, for any integer $x \neq 0$. Also, no integer c satisfies $\mathit{isGCD}(0, 0, c)$ and hence, when $\mathit{isGCD}(x, y, c)$ holds for some integer c, either $x \neq 0$ or $y \neq 0$.

$$\dfrac{\alpha \qquad \dfrac{\dfrac{\text{pred. calculus + algebra}}{\mathit{isGCD}(x,y,c)\wedge x \geq y \rightarrow \mathit{isGCD}(x-y,y,c)\wedge x-y \geq 0} \qquad \beta \qquad \dfrac{\text{pred. calculus + algebra}}{\mathit{isGCD}(x,y,c)\wedge x \geq 0 \rightarrow \mathit{isGCD}(x,y,c)}}{\{\mathit{isGCD}(x,y,c)\wedge x \geq y\}\ \text{S3}\ \{\mathit{isGCD}(x,y,c)\}}\ \text{(Cons.)}}{\{\mathit{isGCD}(x,y,c)\}\ \text{S1; S3}\ \{\mathit{isGCD}(x,y,c)\}}\ \text{(Seq.)}$$

α:

$$\dfrac{\dfrac{\mathit{isGCD}(x,y,c)\wedge x<y \rightarrow \mathit{isGCD}(y,x,c)\wedge y \geq x \qquad \gamma}{\{\mathit{isGCD}(x,y,c)\wedge x<y\}\ \text{S2}\ \{\mathit{isGCD}(x,y,c)\wedge x \geq y\}}\ \text{(S. Pre.)} \qquad \dfrac{\text{pred. calculus + algebra}}{\mathit{isGCD}(x,y,c)\wedge \neg(x<y) \rightarrow \mathit{isGCD}(x,y,c)\wedge x \geq y}}{\{\mathit{isGCD}(x,y,c)\}\ \text{S1}\ \{\mathit{isGCD}(x,y,c)\wedge x \geq y\}}\ \text{(If-Then)}$$

β:

$$\dfrac{}{\{\mathit{isGCD}(x-y,y,c)\wedge x-y \geq 0\}\ \text{S3: } x := x-y\ \{\mathit{isGCD}(x,y,c)\wedge x \geq 0\}}\ \text{(Assign.)}$$

γ:

$$\dfrac{}{\{\mathit{isGCD}(y,x,c)\wedge y \geq x\}\ \text{S2: } x,y := y,x\ \{\mathit{isGCD}(x,y,c)\wedge x \geq y\}}\ \text{(Assign.)}$$

□

   (b) (10 points)

   $\{n \geq 1\}$
   S1: $g, p := 0, n$;
   S2: **while** B: $p \geq 2$ **do**
       S3: $g, p := g + 1, p - 1$
   od
   $\{g = n - 1\}$

   Solution.

$$\dfrac{\dfrac{n \geq 1 \to 0 = 0 \wedge n = n \wedge n \geq 1 \qquad \alpha}{\{n \geq 1\}\ \text{S1}\ \{g = 0 \wedge p = n \wedge n \geq 1\}}\text{(S. Pre.)} \qquad \dfrac{g = 0 \wedge p = n \wedge n \geq 1 \to p > 0 \wedge p + g = n \qquad \beta \qquad \gamma}{\{g = 0 \wedge p = n \wedge n \geq 1\}\ \text{S2}\ \{g = n - 1\}}\text{(Cons.)}}{\{n \geq 1\}\ \text{S1; S2}\ \{g = n - 1\}}\text{(Seq.)}$$

α :

$$\dfrac{}{\{0 = 0 \wedge n = n \wedge n \geq 1\}\ \text{S1: } g, p := 0, n\ \{g = 0 \wedge p = n \wedge n \geq 1\}}\text{(Assign.)}$$

β :

$$\dfrac{\dfrac{\beta_1 \qquad \dfrac{}{\{\ p - 1 > 0 \wedge (p - 1) + (g + 1) = n\ \}\ g, p := g + 1, p - 1\ \{\ p > 0 \wedge p + g = n\ \}}\text{(Assign)}}{\{\ p > 0 \wedge p + g = n \wedge p \geq 2\ \}\ g, p := g + 1, p - 1\ \{\ p > 0 \wedge p + g = n\ \}}\text{(SP)}}{\{\ p > 0 \wedge p + g = n\ \}\ \text{S2: } \textbf{while}\ p \geq 2\ \textbf{do}\ g, p := g + 1, p - 1\ \textbf{od}\ \{\ p > 0 \wedge p + g = n \wedge \neg(p \geq 2)\ \}}\text{(While)}$$

β_1 :

$$\dfrac{\text{pred. calculus + algebra}}{p > 0 \wedge p + g = n \wedge p \geq 2 \to p - 1 > 0 \wedge (p - 1) + (g + 1) = n}$$

γ :

$$\dfrac{\text{pred. calculus + algebra}}{p > 0 \wedge p + g = n \wedge \neg(p \geq 2) \to g = n - 1}$$

$\square$

(c) (20 points) For this program, prove its total correctness.

$\{n > 0\}$
S1: $x, y := n, 0$;
S2: **while** B: $x > 0$ **do**
  S3: $y := y + 2 \times x - 1$;
  S4: $x := x - 1$
od
$\{y = n^2\}$

Solution.

$$\dfrac{\dfrac{\dfrac{\text{pred. calculus + algebra}}{n > 0 \to n = n \wedge 0 = 0 \wedge n > 0} \qquad \alpha}{\{n > 0\}\ \text{S1}\ \{x = n \wedge y = 0 \wedge n > 0\}}\text{(S. Pre.)} \qquad \dfrac{\dfrac{\text{pred. calculus + algebra}}{x = n \wedge y = 0 \wedge n > 0 \to 0 \leq x \leq n \wedge y = n^2 - x^2 \wedge n > 0} \qquad \beta \qquad \gamma}{\{x = n \wedge y = 0 \wedge n > 0\}\ \text{S2}\ \{y = n^2\}}\text{(Cons.)}}{\{n > 0\}\ \text{S1; S2}\ \{y = n^2\}}\text{(Seq.)}$$

α :

$$\dfrac{}{\{n = n \wedge 0 = 0 \wedge n > 0\}\ \text{S1: } x, y := n, 0\ \{x = n \wedge y = 0 \wedge n > 0\}}\text{(Assign.)}$$

β :

$$\dfrac{\beta_1 \qquad \dfrac{\dfrac{\text{pred. calculus + algebra}}{P \wedge x > 0 \wedge x = Z \to x = Z} \qquad \beta_2}{\{P \wedge x > 0 \wedge x = Z\}\ \text{S3; S4}\ \{x < Z\}}\text{(S. Pre.)} \qquad \dfrac{\text{pred. calculus + algebra}}{P \wedge (x > 0) \to x \geq 0}}{\{P : 0 \leq x \leq n \wedge y = n^2 - x^2 \wedge n > 0\}\ \text{S2}\ \{P \wedge \neg(x > 0)\}}\text{(While: simply total)}$$

β_1 :

$$\cfrac{\cfrac{\cfrac{\text{pred. calculus + algebra}}{P \wedge x > 0 \rightarrow 0 \leq x \leq n \wedge y + 2x - 1 = n^2 - (x-1)^2 \wedge n > 0 \wedge x > 0} \quad \beta_{11}}{\{P \wedge x > 0\}\ \text{S3}\ \{0 \leq x \leq n \wedge y = n^2 - (x-1)^2 \wedge n > 0 \wedge x > 0\}}\,\text{(S. Pre.)} \quad \beta_{12}}{\{P \wedge x > 0\}\ \text{S3; S4}\ \{P\}}\,\text{(Seq.)}$$

β_{11} :

$$\frac{}{\{0 \leq x \leq n \wedge y + 2x - 1 = n^2 - (x-1)^2 \wedge n > 0 \wedge x > 0\}\ \text{S3}\ \{0 \leq x \leq n \wedge y = n^2 - (x-1)^2 \wedge n > 0 \wedge x > 0\}}\,\text{(Assign.)}$$

β_{12} :

$$\cfrac{\beta_{121} \quad \cfrac{}{\{0 \leq x - 1 \leq n \wedge y = n^2 - (x-1)^2 \wedge n > 0 \wedge x - 1 \geq 0\}\ \text{S4}\ \{P \wedge x \geq 0\}}\,\text{(Assign.)} \quad \cfrac{\text{pred. calculus + algebra}}{P \wedge x \geq 0 \rightarrow P}}{\{0 \leq x \leq n \wedge y = n^2 - (x-1)^2 \wedge n > 0 \wedge x > 0\}\ \text{S4}\ \{P\}}\,\text{(Cons.)}$$

β_{121} :

$$\frac{\text{pred. calculus + algebra}}{0 \leq x \leq n \wedge y = n^2 - (x-1)^2 \wedge n > 0 \wedge x > 0 \rightarrow 0 \leq x - 1 \leq n \wedge y = n^2 - (x-1)^2 \wedge n > 0 \wedge x - 1 \geq 0}$$

β_2 :

$$\cfrac{\cfrac{}{\{x = Z\}\ \text{S3: } y := y + 2 \times x - 1\ \{x = Z\}}\,\text{(Assign.)} \quad \cfrac{\cfrac{\text{pred. calculus + algebra}}{\{x = Z \rightarrow x - 1 < Z\}} \quad \cfrac{}{\{x - 1 < Z\}\ \text{S4: } x := x - 1\ \{x < Z\}}\,\text{(Assign.)}}{\{x = Z\}\ \text{S4}\ \{x < Z\}}\,\text{(S. Pre.)}}{\{x = Z\}\ \text{S3; S4}\ \{x < Z\}}\,\text{(Seq.)}$$

γ :

$$\frac{\text{pred. calculus + algebra}}{P \wedge \neg(x > 0) \rightarrow y = n^2}$$

□

2. (30 %) Below is a program that finds the minimum and the maximum elements of an array of n (assumed to be positive and even) integers. The elements of an array are indexed from 1 through n.

```
if (a[1] < a[2]) then
  min := a[1];
  max := a[2]
else
  min := a[2];
  max := a[1]
fi;

i := 3;
while (i<=n) do
  if (a[i] < a[i+1]) then
    if (a[i] < min) then
      min := a[i]
    fi;
    if (a[i+1] > max) then
```

```
      max := a[i+1];
    fi
  else
    if (a[i+1] < min) then
      min := a[i+1]
    fi;
    if (a[i] > max) then
      max := a[i]
    fi
  fi;
  i := i + 2;
od;
```

Annotate the program into a *standard* proof outline, showing clearly the partial correctness of the program; a standard proof outline is essentially an annotated program where every statement is preceded by a pre-condition and the entire program is followed by a post-condition.

Solution. Let $isMin(m, a, i)$ denote that m is the minimum element in $a[1..i]$ and $isMax(M, a, i)$ denote that M is the maximum element in $a[1..i]$. In particular, $isMin(\_, a, 0)$ and $isMax(\_, a, 0)$ both hold, as $a[1..0]$ denotes the empty array. Let $odd(i)$ denote that i is odd.

```
// assume n is positive and even, which is preserved by the code and
// will be omitted later
if (a[1] < a[2]) then
// $isMin(a[1], a, 2) \wedge isMax(a[2], a, 2)$
  min := a[1];
// $isMin(min, a, 2) \wedge isMax(a[2], a, 2)$
  max := a[2]
else
// $isMin(a[2], a, 2) \wedge isMax(a[1], a, 2)$
  min := a[2];
// $isMin(min, a, 2) \wedge isMax(a[1], a, 2)$
  max := a[1]
fi;
// $isMin(min, a, 2) \wedge isMax(max, a, 2)$

i := 3;
// inv: $(3 \le i \le n+1) \wedge odd(i) \wedge isMin(min, a, i-1) \wedge isMax(max, a, i-1)$
while (i<=n) do
  // $(3 \le i \le n) \wedge odd(i) \wedge isMin(min, a, i-1) \wedge isMax(max, a, i-1)$
  if (a[i] < a[i+1]) then
    // $(3 \le i \le n) \wedge odd(i) \wedge (a[i] < a[i+1]) \wedge isMin(min, a, i-1) \wedge isMax(max, a, i-1)$
    if (a[i] < min) then
    // $(3 \le i \le n) \wedge odd(i) \wedge (a[i] < a[i+1]) \wedge isMin(a[i], a, i+1) \wedge isMax(max, a, i-1)$
      min := a[i]
    fi;
    // $(3 \le i \le n) \wedge odd(i) \wedge (a[i] < a[i+1]) \wedge isMin(min, a, i+1) \wedge isMax(max, a, i-1)$
    if (a[i+1] > max) then
```

```
        // $(3 \le i \le n) \wedge odd(i) \wedge (a[i] < a[i+1]) \wedge isMin(min, a, i+1) \wedge$
        // $isMax(a[i+1], a, i+1)$
        max := a[i+1];
      fi
  else
      // $(3 \le i \le n) \wedge odd(i) \wedge (a[i] \ge a[i+1]) \wedge isMin(min, a, i-1) \wedge isMax(max, a, i-1)$
      if (a[i+1] < min) then
        // $(3 \le i \le n) \wedge odd(i) \wedge (a[i] \ge a[i+1]) \wedge isMin(a[i+1], a, i+1) \wedge$
        // $isMax(max, a, i-1)$
        min := a[i+1]
      fi;
      // $(3 \le i \le n) \wedge odd(i) \wedge (a[i] \ge a[i+1]) \wedge isMin(min, a, i+1) \wedge isMax(max, a, i-1)$
      if (a[i] > max) then
        // $(3 \le i \le n) \wedge odd(i) \wedge (a[i] \ge a[i+1]) \wedge isMin(min, a, i+1) \wedge$
        // $isMax(a[i], a, i+1)$
        max := a[i]
      fi
  fi;
  // $(3 \le i \le n) \wedge odd(i) \wedge isMin(min, a, i+1) \wedge isMax(max, a, i+1)$
  i := i + 2;
  // $(3 \le i \le n+1) \wedge odd(i) \wedge isMin(min, a, i-1) \wedge isMax(max, a, i-1)$
od;
// $isMin(min, a, n) \wedge isMax(max, a, n)$
```

□

3. (30 points) Given a sequence x_1, x_2, $\cdots$, x_n of real numbers (not necessarily positive), a maximum subsequence x_i, x_{i+1}, $\cdots$, x_j is a subsequence of consecutive elements from the given sequence such that the sum of the numbers in the subsequence is maximum over all subsequences of consecutive elements. Below is a program that determines the sum of such a sequence.

```
Global_Max := 0;
Suffix_Max := 0;
i := 1;
while (i<=n) do
   if x[i] + Suffix_Max > Global_Max then
      Suffix_Max := Suffix_Max + x[i];
      Global_Max := Suffix_Max
   else
     if x[i] + Suffix_Max > 0 then
       Suffix_Max := Suffix_Max + x[i]
     else Suffix_Max := 0
     fi
   fi;
   i := i + 1
od;
```

Annotate the program into a standard proof outline, showing clearly the partial correctness of the program.

Solution. Let $\mathit{isMS}(s, x, i)$ denote that s is the sum of the maximum subsequence in $x[1..i]$ and $\mathit{isMSX}(s, x, i)$ denote that s is the sum of the maximum subsequence that is also a suffix in $x[1..i]$. In particular, $\mathit{isMS}(0, x, 0)$ and $\mathit{isMSX}(0, x, 0)$ both hold, as $x[1..0]$ denotes the empty sequence. To shorten formulae, we denote `Global_Max` and `Suffix_Max` respectively by G_M and S_M in all assertions.

```
// assume n ≥ 1, which is preserved by the code and will be omitted later
Global_Max := 0;
// isMS(G_M, x, 0)
Suffix_Max := 0;
// isMS(G_M, x, 0) ∧ isMSX(S_M, x, 0)
i := 1;
// inv: (1 ≤ i ≤ n + 1) ∧ isMS(G_M, x, i − 1) ∧ isMSX(S_M, x, i − 1)
while i <= n do
    // (1 ≤ i ≤ n) ∧ isMS(G_M, x, i − 1) ∧ isMSX(S_M, x, i − 1)
    if x[i] + Suffix_Max > Global_Max then
        // (1 ≤ i ≤ n) ∧ isMS(x[i] + S_M, x, i) ∧ isMSX(x[i] + S_M, x, i)
        Suffix_Max := Suffix_Max + x[i];
        // (1 ≤ i ≤ n) ∧ isMS(S_M, x, i) ∧ isMSX(S_M, x, i)
        Global_Max := Suffix_Max
    else
          // (1 ≤ i ≤ n) ∧ isMS(G_M, x, i) ∧ isMSX(S_M, x, i − 1)
          if x[i] + Suffix_Max > 0 then
              // (1 ≤ i ≤ n) ∧ isMS(G_M, x, i) ∧ isMSX(x[i] + S_M, x, i)
              Suffix_Max := Suffix_Max + x[i]
          else
              // (1 ≤ i ≤ n) ∧ isMS(G_M, x, i) ∧ isMSX(0, x, i)
              Suffix_Max := 0;
          fi
    fi;
    // (1 ≤ i ≤ n) ∧ isMS(G_M, x, i) ∧ isMSX(S_M, x, i)
    i := i + 1
od;
// isMS(G_M, x, i−1) ∧ isMSX(S_M, x, i−1) ∧ i = n+1(implying isMS(G_M, x, n))
```

□