Homework Assignment #4

Due Time/Date

2:20PM Wednesday, November 9, 2022. Late submission will be penalized by 20% for each working day overdue.

How to Submit

Please use a word processor or scan hand-written answers to produce a single PDF file. Name your file according to this pattern: "b097050xx-hw4". Upload the PDF file to the NTU COOL site for Software Specification and Verification 2022. You may discuss the problems with others, but copying answers is strictly forbidden.

Problems

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)

   ```
   $\{true\}$
   if $x < y$ then $x, y := y, x$ fi
   $\{x \geq y\}$
   ```

   (b) (10 points)

   ```
   $\{g = 0 \wedge p = n \wedge n \geq 1\}$
   while $p \geq 2$ do
       $g, p := g + 1, p - 1$
   od
   $\{g = n - 1\}$
   ```

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

   ```
   $\{y > 0 \wedge (x \equiv m \pmod{y})\}$
   while $x \geq y$ do
       $x := x - y$
   od
   $\{(x \equiv m \pmod{y}) \wedge x < y\}$
   ```

2. (20 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;
for i := 1 to 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
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.

3. (40 points) Given a directed graph represented by an $n \times n$ adjacency matrix (named `Know[1..n, 1..n]`), the following program determines whether there exists an i (the sink or "celebrity" of the graph) such that all the entries in the i-th column (except for the ii-th entry) are 1, and all the entries in the i-th row (except for the ii-th entry) are 0.

```
i, j, next := 1, 2, 3;
while next <= n+1 do
  if Know[i,j] then i := next
  else j := next;
  next := next + 1;
od
if i = n+1 then candidate := j
else candidate := i;

wrong := false;
k := 1;
Know[candidate,candidate] := false;
while not wrong and k <= n do
  if Know[candidate,k] then wrong := true;
  if not Know[k,candidate] then
    if candidate <> k then wrong := true;
  k := k + 1;
od
if not wrong then celebrity := candidate
else celebrity := 0;
```

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