Bhavesh Rajesh Talreja (A20516822)
CS 536 HW5

1. Minimal and Full Proof Outlines

Task 1.1

Given Program:

```
                                 {x ≥ 0}
i := 0̄;
{inv i ≤ x ∧ x = i!}
while i < x do
   x := x * i;
   i := i + 1
od
                                 {∃k.x = k!}
```

Ans.

We are asked to convert this minimal proof outline to a full proof outline.

For this we will use the set of rules/algorithm learnt in class.

To begin with, we will use rule 9:
Add conditions to a loop based on the invariant:

$$\{\text{inv } P\} \text{ while } e \text{ do } S \text{ od}$$
$$\text{becomes}$$
$$\{\text{inv } P\} \text{ while } e \text{ do } \{P \wedge e\}\ S\ \{P\} \text{ od } \{P \wedge \neg e\}$$

```
                                 {x ≥ 0}
i := 0̄;
{inv i ≤ x ∧ x = i!}
while i < x do                   {i ≤ x ∧ x = i! ∧ i < x}
   x := x * i;
   i := i + 1                    {i ≤ x ∧ x = i!}
od                               {i ≤ x ∧ x = i! ∧ i ≥ x}
                                 {∃k.x = k!}
```

Now, before proceeding ahead, we have a proof obligation which we need to try to proof.
In our case, we will try to prove $\{i \leq x \wedge x = i! \wedge i \geq x\} \Longrightarrow \{\exists k.x = k!\}$

Explanation: From the above, we can see that $i \leq x$ and $i \geq x$, i.e., $i = x$.
$\{i \leq x \wedge x = i! \wedge i \geq x\} \Longrightarrow \{i = x \wedge x = i!\}$

This says that there is an value of x which is equal to i, and x = i!. Exactly what $\{\exists k.x = k!\}$ says.

So, $\{i \leq x \land x = i! \land i \geq x\} \Longrightarrow \{\exists k.x = k!\}$

Since, we proved the proof obligation, we can write the proof outline as:

```
                                  {x ≥ 0}
i := 0̄;
{inv i ≤ x ∧ x = i!}
while i < x do                    {i ≤ x ∧ x = i! ∧ i < x}
  x := x * i;
  i := i + 1                      {i ≤ x ∧ x = i!}
od                                {i ≤ x ∧ x = i! ∧ i ≥ x}
                                  ⟹ {∃k.x = k!}
```

Next, we will use rule 1:

Prepend {wlp(x := e, Q)} to x := e {Q}.

```
                                  {x ≥ 0}
i := 0̄;
{inv i ≤ x ∧ x = i!}
while i < x do                    {i ≤ x ∧ x = i! ∧ i < x}
  x := x * i;                     {i+1 ≤ x ∧ x = (i+1)!}
  i := i + 1                      {i ≤ x ∧ x = i!}
od                                {i ≤ x ∧ x = i! ∧ i ≥ x}
                                  ⟹ {∃k.x = k!}
```

Again, we will use rule 1:

Prepend {wlp(x := e, Q)} to x := e {Q}.

```
                                  {x ≥ 0}
i := 0̄;
{inv i ≤ x ∧ x = i!}
while i < x do                    {i ≤ x ∧ x = i! ∧ i < x}   {i+1 ≤ x*i ∧ x = (i+1)!}
  x := x * i;                     {i+1 ≤ x ∧ x = (i+1)!}
  i := i + 1                      {i ≤ x ∧ x = i!}
od                                {i ≤ x ∧ x = i! ∧ i ≥ x}
                                  ⟹ {∃k.x = k!}
```

We again come across a proof obligation that we need to prove before we can proceed further.

$\{i \le x \land x = i! \land i < x\} \quad \{i+1 \le x*i \land x = (i+1)!\}$

Explanation: Since $i \le x$ and $i < x$ means i is at the maximum equal to x or less than x. So, it would be correct to say that i+1 will be less than or equal to x*i.

We are saying that $i+1 \le x*i$. But we cannot say anything about $x = (i+1)!$
So, $x = i! \nRightarrow x = (i+1)!$
i.e., $\{i \le x \land x = i! \land i < x\} \nRightarrow \{i+1 \le x*i \land x = (i+1)!\}$
This proof obligation cannot be proved.

Since, we were not able to prove the proof obligation, we can write the proof outline as:

```
                                  $\{x \ge 0\}$
i := $\bar{0}$;
{inv $i \le x \land x = i!$}
while i < x do                    $\{i \le x \land x = i! \land i < x\} \nRightarrow \{i+1 \le x*i \land x = (i+1)!\}$
   x := x * i;                    $\{i+1 \le x \land x = (i+1)!\}$
   i := i + 1                     $\{i \le x \land x = i!\}$
od                                $\{i \le x \land x = i! \land i \ge x\}$
                                  $\Longrightarrow \{\exists k.x = k!\}$
```

Now we will use rule 1, i.e., wlp to propagate the postcondition of the loop body backwards.

```
                                  $\{x \ge 0\}$          $\{0 \le x \land x = 0!\}$
i := $\bar{0}$;                   $\{i \le x \land x = i!\}$
{inv $i \le x \land x = i!$}
while i < x do                    $\{i \le x \land x = i! \land i < x\} \nRightarrow \{i+1 \le x*i \land x = (i+1)!\}$
   x := x * i;                    $\{i+1 \le x \land x = (i+1)!\}$
   i := i + 1                     $\{i \le x \land x = i!\}$
od                                $\{i \le x \land x = i! \land i \ge x\}$
                                  $\Longrightarrow \{\exists k.x = k!\}$
```

We again come across a proof obligation that we need to prove before we can proceed further.

$\{x \ge 0\} \qquad \{0 \le x \land x = 0!\}$

Explanation: $\{x \ge 0\} \Longrightarrow \{x \ge 0 \land x = 1\} \Longrightarrow \{x \ge 0 \land x = 0!\} \Longrightarrow \{0 \le x \land x = 0!\}$

This proof obligation is proved.

Since, we were able to prove the proof obligation, we can write the proof outline as:

```
                                  {x ≥ 0} ⟹ {0 ≤ x ∧ x = 0!}
i := 0̄;                           {i ≤ x ∧ x = i!}
{inv i ≤ x ∧ x = i!}
while i < x do                    {i ≤ x ∧ x = i! ∧ i < x} ⇏{i+1 ≤ x*i ∧ x = (i+1)!}
  x := x * i;                     {i+1 ≤ x ∧ x = (i+1)!}
  i := i + 1                      {i ≤ x ∧ x = i!}
od                                {i ≤ x ∧ x = i! ∧ i ≥ x}
                                  ⟹ {∃k.x = k!}
```

This is the full proof outline with all the proof obligations (we were not able to prove one of the three proof obligations we came across while converting the above minimal proof outline to full proof outline)

Task 1.2

Given Program:

```
                                  {x ≥ 0}
i := 0̄;
r := 1̄;
while i < x do
  i := i + 1;
  r := r * i;
od
                                  {r = x!}
```

Ans.

Here we must provide an invariant for this program. The invariant that worked for me is

{inv i ≤ x ∧ r = i!}

We are asked to convert this minimal proof outline to a full proof outline.

For this we will use the set of rules/algorithm learnt in class.

To begin with, we will use rule 9:
Add conditions to a loop based on the invariant:

{inv P} while e do S od
becomes

Bhavesh Rajesh Talreja (A20516822)
CS 536 HW5

$\{inv\ P\}$ while e do $\{P \wedge e\}$ S $\{P\}$ od $\{P \wedge \neg e\}$

```
                                        {x ≥ 0}
i := $\overline{0}$;
r := $\overline{1}$;
{inv i ≤ x ∧ r = i!}
while i < x do                          {i ≤ x ∧ r = i! ∧ i < x}
   i := i + 1;
   r := r * i;                          {i ≤ x ∧ r = i!}
od                                      {i ≤ x ∧ r = i! ∧ i ≥ x}
                                        {r = x!}
```

Now, before proceeding ahead, we have a proof obligation which we need to try to proof.
In our case, we will try to prove $\{i \le x \wedge r = i! \wedge i \ge x\} \Longrightarrow \{r = x!\}$

Explanation: From the above, we can see that $i \le x$ and $i \ge x$, i.e., $i = x$.
$\{i \le x \wedge r = i! \wedge i \ge x\} \Longrightarrow \{i = x \wedge r = i!\} \Longrightarrow \{r = x!\}$

So, $\{i \le x \wedge r = i! \wedge i \ge x\} \Longrightarrow \{r = x!\}$

Since, we proved the proof obligation, we can write the proof outline as:

```
                                        {x ≥ 0}
i := $\overline{0}$;
r := $\overline{1}$;
{inv i ≤ x ∧ r = i!}
while i < x do                          {i ≤ x ∧ r = i! ∧ i < x}
   i := i + 1;
   r := r * i;                          {i ≤ x ∧ r = i!}
od                                      {i ≤ x ∧ r = i! ∧ i ≥ x}
                                        ⟹ {r = x!}
```

Next, we will use rule 1:

Prepend {wlp(x := e, Q)} to x := e {Q}.

```
                                        {x ≥ 0}
i := $\overline{0}$;
r := $\overline{1}$;
{inv i ≤ x ∧ r = i!}
while i < x do                          {i ≤ x ∧ r = i! ∧ i < x}
   i := i + 1;                          {i ≤ x ∧ r * i = i!}
   r := r * i;                                 {i ≤ x ∧ r = i!}
od                                             {i ≤ x ∧ r = i! ∧ i ≥ x}
                                               ⟹ {r = x!}
```

Again, we will use rule 1:

Prepend {wlp(x := e, Q)} to x := e {Q}.

```
                                               {x ≥ 0}
i := 0̄;
r := 1̄;
{inv i ≤ x ∧ r = i!}
while i < x do                                 {i ≤ x ∧ r = i! ∧ i < x}    {i+1 ≤ x ∧ r * (i+1) = (i+1)!}
   i := i + 1;                                 {i ≤ x ∧ r * i = i!}
   r := r * i;                                 {i ≤ x ∧ r = i!}
od                                             {i ≤ x ∧ r = i! ∧ i ≥ x}
                                               ⟹ {r = x!}
```

We again come across a proof obligation that we need to prove before we can proceed further.

$\{i \leq x \wedge r = i! \wedge i < x\} \quad \{i+1 \leq x \wedge r * (i+1) = (i+1)!\}$

Explanation: Since $i < x$, so it would be correct to say that $i+1 \leq x$.

$\{i \leq x \wedge r = i! \wedge i < x\} \Longrightarrow \{i < x \wedge r = i!\}$ …(1)

Using the definition of Factorial,
$m! = m * (m-1)!$
i.e., $m! * (m+1) = (m+1)!$ …(2)

From (1) and (2), we can say,

$\{i \leq x \wedge r = i! \wedge i < x\} \Longrightarrow \{i < x \wedge r = i!\} \Longrightarrow \{i+1 \leq x \wedge r * (i+1) = (i+1)!\}$

So, $\{i \leq x \wedge r = i! \wedge i < x\} \Longrightarrow \{i+1 \leq x \wedge r * (i+1) = (i+1)!\}$
Since, we were able to prove the proof obligation, we can write the proof outline as:

```
                                               {x ≥ 0}
i := 0̄;
r := 1̄;
{inv i ≤ x ∧ r = i!}
while i < x do                                 {i ≤ x ∧ r = i! ∧ i < x} ⟹ {i+1 ≤ x ∧ r * (i+1) = (i+1)!}
```

```
   i := i + 1;                                {i ≤ x ∧ r * i = i!}
   r := r * i;                                {i ≤ x ∧ r = i!}
od                                            {i ≤ x ∧ r = i! ∧ i ≥ x}
                                              ⟹ {r = x!}
```

Now we will use rule 1 multiple times on the assignment statements before the loop, and wlp to propagate the postcondition of the loop body backwards.

```
                                              {x ≥ 0}          {0 ≤ x ∧ 1 = 0!}
i := 0̅;                                       {i ≤ x ∧ 1 = i!}
r := 1̅;                                       {i ≤ x ∧ r = i!}
{inv i ≤ x ∧ r = i!}
while i < x do                                {i ≤ x ∧ r = i! ∧ i < x} ⟹ {i+1 ≤ x ∧ r * (i+1) = (i+1)!}
   i := i + 1;                                {i ≤ x ∧ r * i = i!}
   r := r * i;                                {i ≤ x ∧ r = i!}
od                                            {i ≤ x ∧ r = i! ∧ i ≥ x}
                                              ⟹ {r = x!}
```

We again come across a proof obligation that we need to prove before we can proceed further.

{x ≥ 0}   {0 ≤ x ∧ 1 = 0!}

Explanation: {x ≥ 0} ⟹ {x ≥ 0 ∧ 1 = 1} ⟹ {x ≥ 0 ∧ 1 = 0!} ⟹ {0 ≤ x ∧ 1 = 0!}

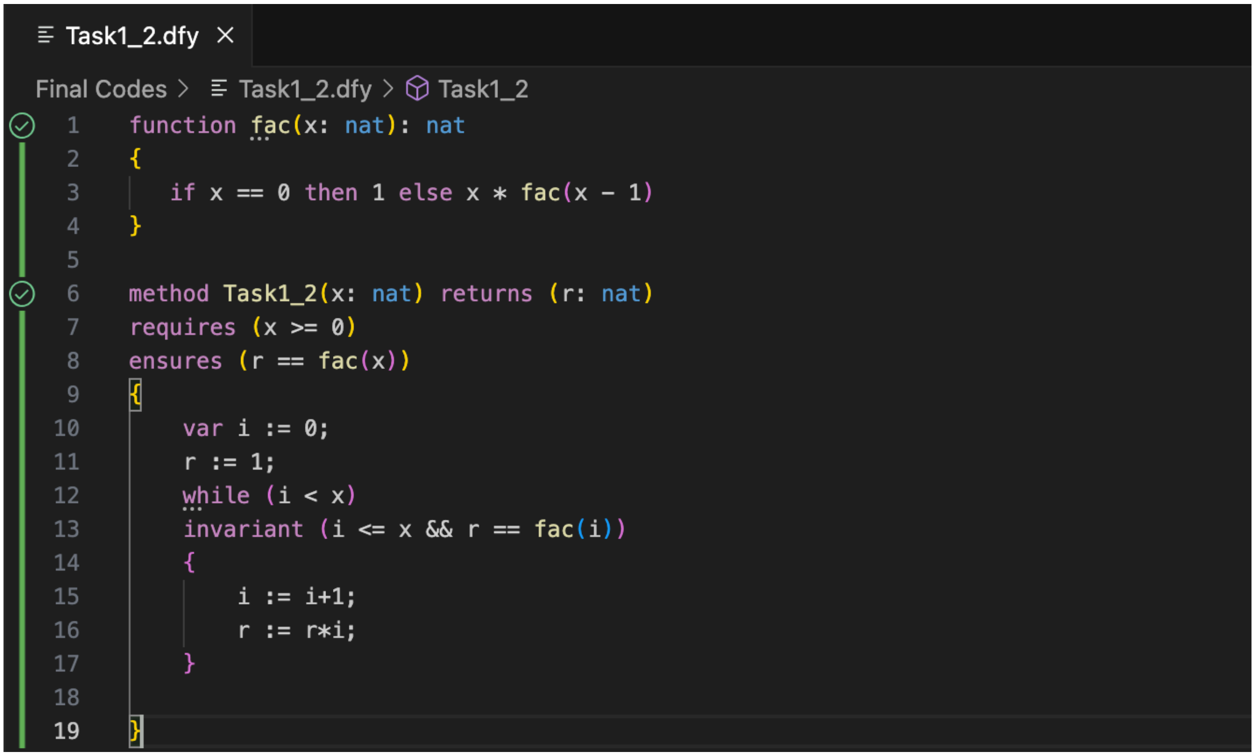
This proof obligation is proved.

Since, we were able to prove the proof obligation, we can write the proof outline as:

```
                                              {x ≥ 0} ⟹ {0 ≤ x ∧ 1 = 0!}
i := 0̅;                                       {i ≤ x ∧ 1 = i!}
r := 1̅;                                       {i ≤ x ∧ r = i!}
{inv i ≤ x ∧ r = i!}
while i < x do                                {i ≤ x ∧ r = i! ∧ i < x} ⟹ {i+1 ≤ x ∧ r * (i+1) = (i+1)!}
   i := i + 1;                                {i ≤ x ∧ r * i = i!}
   r := r * i;                                {i ≤ x ∧ r = i!}
od                                            {i ≤ x ∧ r = i! ∧ i ≥ x}
                                              ⟹ {r = x!}
```

This is the full proof outline with all the proof obligations (we were able to prove all of the three proof obligations we came across while converting the above minimal proof outline to full proof outline).

Same can be verified from the snip of my trial in Dafny.



```
function fac(x: nat): nat
{
    if x == 0 then 1 else x * fac(x - 1)
}

method Task1_2(x: nat) returns (r: nat)
requires (x >= 0)
ensures (r == fac(x))
{
    var i := 0;
    r := 1;
    while (i < x)
    invariant (i <= x && r == fac(i))
    {
        i := i+1;
        r := r*i;
    }

}
```

Bhavesh Rajesh Talreja (A20516822)
CS 536 HW5

2. Proofs with Loops

Task 2.1

a) Given,
Precondition: T
Postcondition: $\forall i \, \epsilon \, \mathbb{Z}. (0 \ \le i \ < |a|) \ \rightarrow a[i] \, ! \neq r$

Ans.
In this answer, we will write IMP language program:

[T] S [$\forall i \, \epsilon \, \mathbb{Z}. (0 \ \le i \ < |a|) \ \rightarrow a[i] \ \neq r$]

where S ≜

```
r := $\overline{0}$;
if |a| == 0 then
   r := $\overline{0}$
else
   r := a[0];
   i := $\overline{0}$;
   while i < |a| do
        if a[i] $\le$ r then r := a[i] fi;
        i := i + 1
   od;
   r := r – 1
fi
```

b) Given [T] S [$\forall i \, \epsilon \, \mathbb{Z}. (0 \ \le i \ < |a|) \ \rightarrow a[i] \ \neq r$]

where S ≜

```
r := $\overline{0}$;
if |a| == 0 then
   r := $\overline{0}$
else
   r := a[0];
   i := $\overline{0}$;
   while i < |a| do
        if a[i] $\le$ r then r := a[i] fi;
        i := i + 1
   od;
   r := r – 1
fi
```

Bhavesh Rajesh Talreja (A20516822)
CS 536 HW5

Ans. Our task here is to provide correct loop invariant (i.e., it must hold at the beginning of the loop, imply the postcondition at the end of the loop, and be preserved by the loop body).

[T] S [$\forall i \epsilon \mathbb{Z}. (0 \leq i < |a|) \rightarrow a[i] \neq r$]

where S ≜

```
r := $\overline{0}$;
if |a| == 0 then
   r := $\overline{0}$
else
   r := a[0];
   i := $\overline{0}$;
   while i < |a| do
        if a[i] $\leq$ r then r := a[i] fi;
        i := i + 1
   od;
   r := r – 1
fi
```

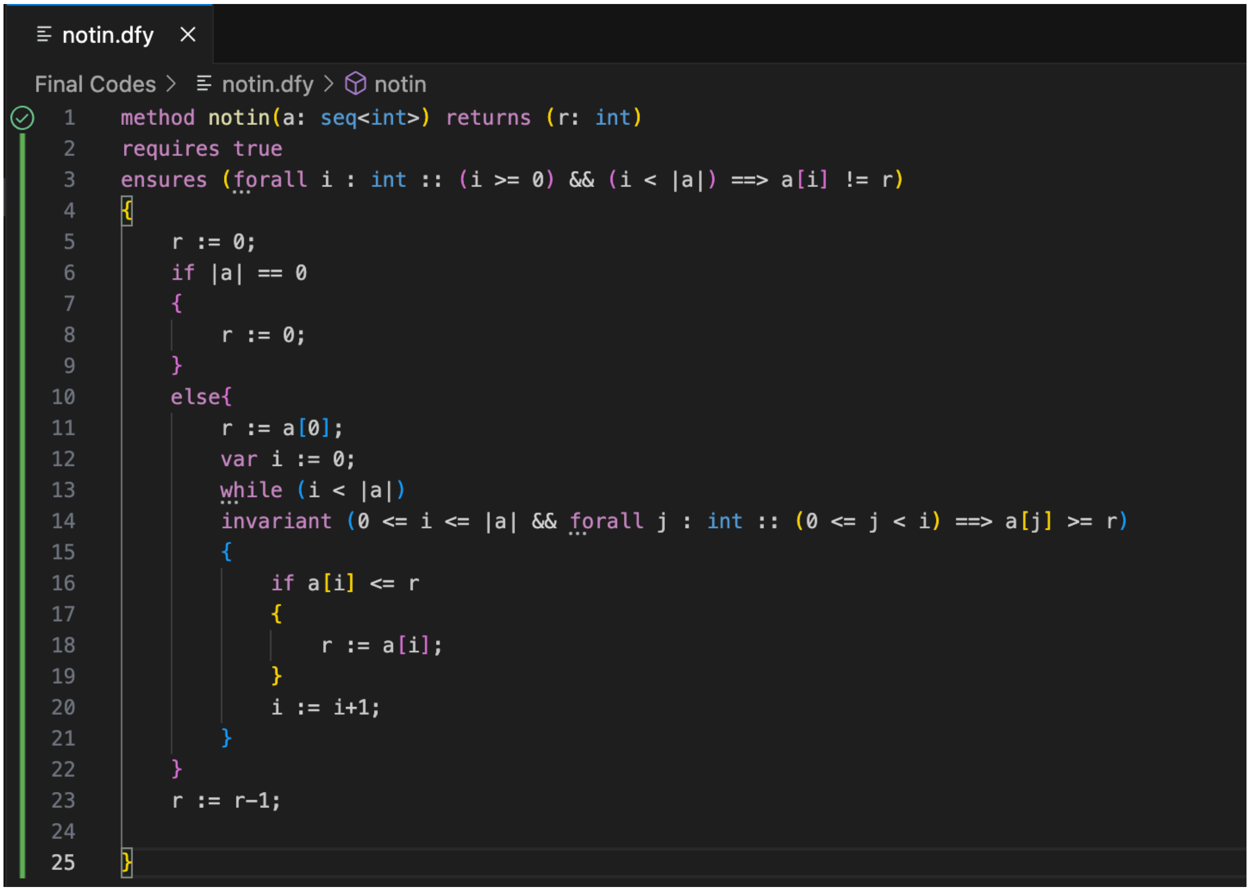
A loop variant for this program can be:

{inv $0 \leq i \leq |a| \wedge \forall j \epsilon \mathbb{Z}. (0 \leq j < i) \rightarrow a[j] \geq r$}

With this loop invariant, the minimal proof outline for this program can be:

```
                                                          [T]
r := $\overline{0}$;
if |a| == 0 then
   r := $\overline{0}$
else
   r := a[0];
   i := $\overline{0}$;
   {inv $0 \leq i \leq |a| \wedge \forall j \epsilon \mathbb{Z}. (0 \leq j < i) \rightarrow a[j] \geq r$}
   while i < |a| do
        if a[i] $\leq$ r then r := a[i] fi;
        i := i + 1
   od;
   r := r – 1
fi
                                                          [$\forall i \epsilon \mathbb{Z}. (0 \leq i < |a|) \rightarrow a[i] \neq r$]
```

This can be verified in Dafny as well:



```
method notin(a: seq<int>) returns (r: int)
requires true
ensures (forall i : int :: (i >= 0) && (i < |a|) ==> a[i] != r)
{
    r := 0;
    if |a| == 0
    {
        r := 0;
    }
    else{
        r := a[0];
        var i := 0;
        while (i < |a|)
        invariant (0 <= i <= |a| && forall j : int :: (0 <= j < i) ==> a[j] >= r)
        {
            if a[i] <= r
            {
                r := a[i];
            }
            i := i+1;
        }
    }
    r := r-1;

}
```

Task 2.2

Given
Precondition: T
Postcondition: b ⟺ numPos(a, 0, |a|) > |a| / 2

Ans. Given program,

```
i := $\overline{0}$;
n := $\overline{0}$;
while i < |a| do
   if a[i] > $\overline{0}$ then n := n + $\overline{1}$ else skip fi;
   i := i + $\overline{1}$;
od
b := n > size(a) / 2
```

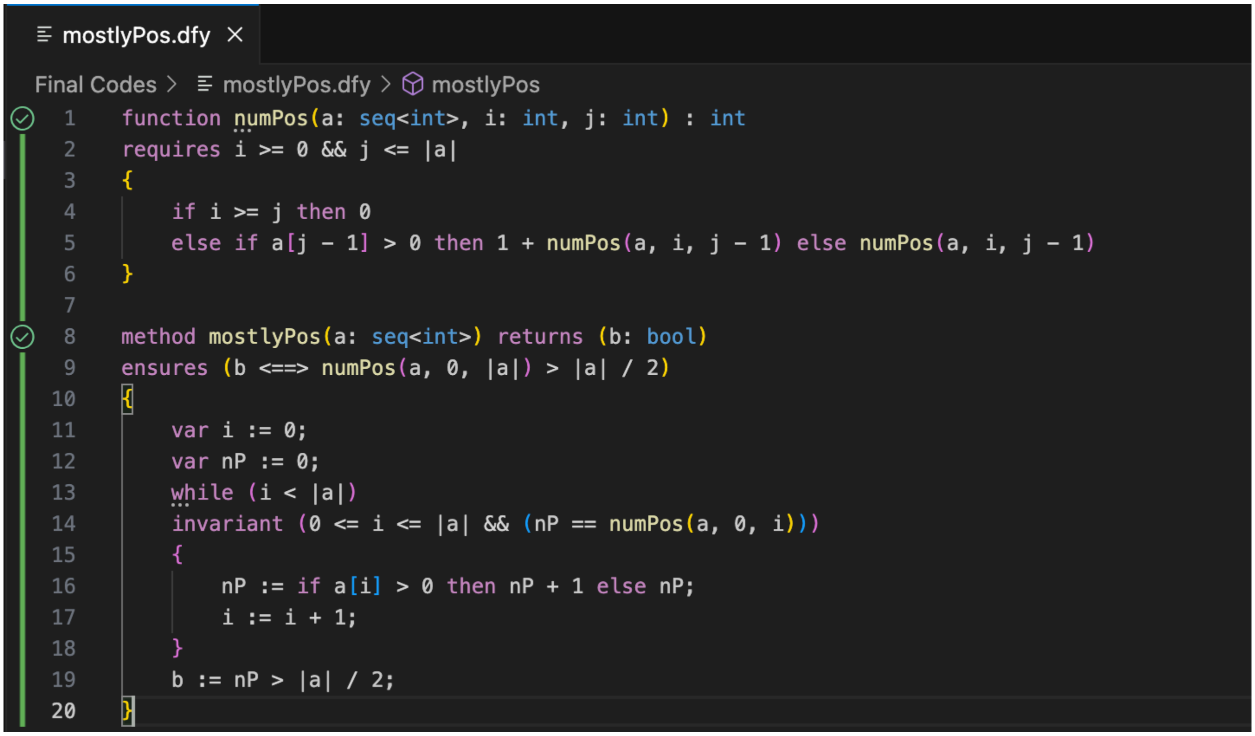
Bhavesh Rajesh Talreja (A20516822)
CS 536 HW5

Our task here is to provide correct loop invariant (i.e., it must hold at the beginning of the loop, imply the postcondition at the end of the loop, and be preserved by the loop body).

A loop variant for this program can be:

$$\{\text{inv } 0 \le i \le |a| \land n = \text{numPos}(a, 0, i)\}$$

This can be verified in Dafny as well.



```
function numPos(a: seq<int>, i: int, j: int) : int
requires i >= 0 && j <= |a|
{
    if i >= j then 0
    else if a[j - 1] > 0 then 1 + numPos(a, i, j - 1) else numPos(a, i, j - 1)
}

method mostlyPos(a: seq<int>) returns (b: bool)
ensures (b <==> numPos(a, 0, |a|) > |a| / 2)
{
    var i := 0;
    var nP := 0;
    while (i < |a|)
    invariant (0 <= i <= |a| && (nP == numPos(a, 0, i)))
    {
        nP := if a[i] > 0 then nP + 1 else nP;
        i := i + 1;
    }
    b := nP > |a| / 2;
}
```

3. One more wrap-up question

Task 3.1

How long (approximately) did you spend on this homework, in total hours of actual working time?

Ans. Totally I spent 18 hours on this assignment.