Assignment 6

Kevin Martin
Winter 2021

Give total correctness proofs of the following programs.

Q1)

```
[ ⊤]
[5 >=0 && 0<=5]                                  Implied, and weakest precondition
x = 5;
[x >=0 && 0<=x]                                  Invariant && variant
while (x > 0) {
      [x>0 && x >=0 && 0<=x=E0]                  Guard && invariant && variant =E0
      [x - 1 >=0 && 0<=x-1<E0]                   Implied, reduce x by 1
       x = x - 1;
      [x >=0 && 0<=x-1<E0]                       Invariant && variant<E0
}
[!(x>0) && x>=0]                                 !guard (exit loop) && invariant
[ x = 0 ]
```

Q2)

```
[ n > 0 ]
[n >= -1 && if odd(n)→n+1, else → n]                          Invariant && variant
while (n > 0) {
                                                              Guard && invariant && variant =E0
      [n > 0 && n >= -1 && 0<(if odd(n)→n+1, else → n)=E0]
      [n - 2 > -1 && 0<=(if odd(n-2)→n-2+1, else → n-2)=E0]
       n = n - 2;                                             Implied, reduce x by 2
      [n >=-1 && 0<=(if odd(n)→n+1, else → n)<E0]
}                                                             Invariant && variant<E0
[!(n > 0) && n>= -1]                                          !guard (exit loop) && invariant
[n + 4 >= 1]                                                  Substitution
n = n + 4;
[ n > 1 ]
```

Q3)

[x > y]

[x+1>=y && 0<=((x-y+1)/2)] — Invariant && variant

```
while (x > y) {
```

[x>y && x+1>=y && 0<=((x-y+1)/2)=E_0] — Guard && invariant && variant =E_0

[x – 1>=y + 1 && 0<=((x – 1 – y+1 +1)/2)<E_0] — Implied, reduce x by 1

```
    x = x - 1;
```

[x + 1>=y + 1 && 0<=((x – y+1 +1)/2)<E_0] — Implied, increase x by 1

```
    y = y + 1;
```

[x + 1>=y && 0<=((x – y+1)/2)<E_0] — Invariant && variant<E_0

```
}
```

[!(x>y) && x+1 >= y] — !guard (exit loop) && invariant

[x<y → x+1=y && (!(x<y)->x=y)] — Implied since guard failed

```
if (x < y)
```

[x + 1 = y] — Increase x by 1, check result

```
    x = x + 1;
```

[x = y]

Else [x = y]

[x = y]

Q4)

[k≥0]

[1=2^0 && 0<=k-0] — Implied

```
    n = 0;
```

[1=2^n && 0<=k-n] — Substitution

```
    x = 1;
```

[x=2^n && 0<=k-n] — Invariant && variant

```
    while (n != k) {
```

[n!=x && x=2^n && 0<=k-n=E_0] — Guard && invariant && variant =E_0

[x + x=2^{n+1} && 0<=k-n+1<E_0] — Implied, increase x by x

```
        x = x + x;
```

[x =2^{n+1} && 0<=k-n+1<E_0] — Invariant && variant<E_0

```
        n = n + 1;
```

[x =2^n && 0<=k-n<E_0] — Invariant && variant<E_0

```
    }
```

[!(n!=k) && x =2^n] — !guard (exit loop) && invariant

[x = 2^k]