CAB203 – Assignment 3

Section 1 Induction

1. Base case: $for\ x = 1, 1^2 \geq 1$.

Inductive case: assuming the statement holds for x, we want to also show that it holds for $x + 1$. We want to show that:

$(x^2 + 1) \geq (x + 1)$

Consider $(x^2 + 1)$

$(x^2 + 1) = x^2 + 2x + 1$

Therefore, since $x^2 \geq x$,

$x^2 + 2x + 1 \geq x + 2x + 1$ which must be true if x is a positive integer.

If the statement is true for $x\ when\ x = 1$, it must be true for $x + 1$

Section 2 Program Correctness

1.

$\{x \in \mathbb{Z}\}$ pre-condition

$\{x \geq 3\}$ pre-condition

2.

$\{x \in \mathbb{Z}\}$ post-condition

$\{x \geq 3\}$ post-condition

$\{y \in \mathbb{Z}\}$ post-condition

$\{y \geq 10\}$ post-condition

$\{y = \mathrm{x}^2 + 1\}$ post-condition

3.

$y = x ** 2$ code block 1

$\{y = x^2\}$ post-condition for block 1

$y = y + 1$ code block 2

$\{y = x^2 + 1\}$ post-condition for block 1

4.

$\{x \in \mathbb{Z}\}$ pre-condition
$\{x \geq 3\}$ pre-condition

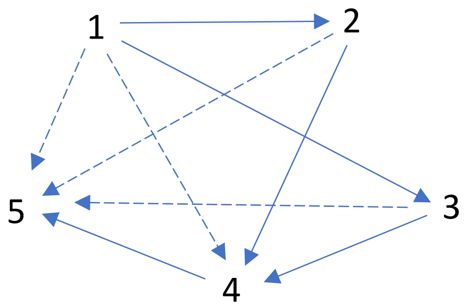
$y = x ** 2$ code block 1
$y = y + 1$ code block 2

$\{x \in \mathbb{Z}\}$ post-condition (Rule 1)
$\{x \geq 3\}$ post-condition (Rule 1)

$\{y \in \mathbb{Z}\}$ post-condition (Assignment rule)
$\{y \geq 10\}$ post-condition (Implication rule)
$\{y = \mathrm{x}^2 + 1\}$ post-condition (Implication rule)

Section 3 Relations, functions and recursion

1.



$$Transitive\ Closure = \{(1{,}2),\ (1{,}3),\ (1{,}4),\ (1{,}5),\ (2{,}4),\ (2{,}5),\ (3{,}4),\ (3{,}5),\ (4{,}5)\ \}$$

2.

```
def equivClasses(S, R):
    equivClasses = {frozenset({y for y in S if (x, y) in R}) for x in S}
    return equivClasses
```

3.

$$\mathrm{f}(x) = \frac{3x + 7}{5}$$

$$y = \frac{3x+7}{5}$$

$$x = \frac{3y+7}{5}$$

$$5x = 3y + 7$$

$$5x - 7 = 3y$$

$$\frac{5x-7}{3} = y$$

$$f^{-1}(x) = \frac{5x-7}{3}$$

4.

The function is not a function in the mathematical since because it is printing out to the screen. This would be considered a side effect.

```
a = myAddition(num1, num2)
b = myAddition(num1, num2)
```

Running it twice would print to the terminal twice.

```
a = myAddition(num1, num2)
b = a
```

Running it in this fashion once would only print to the terminal once.

5.

```
def gcd(x, y):
    if y == 0:
        return x
    if (x >= y and x % y == 0):
        return y
    else:
        return gcd(y, x % y)
```

6.

$$a + 0 = a$$

$$a + S(b) = S(a + b)$$

$1 \text{ can be defined as } S(0)$

$\textit{Since to } a + S(b) = S(a + b)$

$= S(S(S(0)) + S(0))$

$\textit{Since to } a + S(b) = S(a + b)$

$= S(S(S(S(0))) + 0)$

$\textit{Since to } a + 0 = a$

$= S(S(S(S(0))))$ or 4

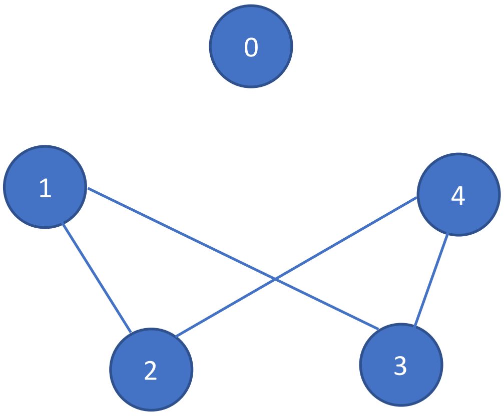
Section 4 Graphs

1. If the root of the tree is E, then the leaves are D and C.

2. **Adjacency List of Graph:**

   = {(A, {C, B}), (B, {A, E, D}), (C, {A}), (D, {B}), (E, {B})}

3. Edges will be (1,2), (1,3), (2, 4), (3,4)



4. $A = \{1, 4\}$

   $B = \{2, 3\}$

5.

$E \rightarrow B \rightarrow A \rightarrow C \rightarrow D$

6.

```
def isBipartite(V, E):
   for u, v in E:
       if(u.isdisjoint(v)):
           u.push(v)
       else:
           return False
   return True
```