ECE 653 Assignment 1

Name: Saride Chandra Mouli

Student ID : 20810347

Email : cmsaride@uwaterloo.ca

1(a):

Empty array doesn’t execute the fault.

X=[] , expected output:0

Actual output:0

1(b):

Here there cant be any case which executes the fault doesn’t lead to error, since its summation without break , it should check all indexes at all times.

1(c):

X=[4,0,-2,0] , Expected output: 4

Actual output: 4

1(d):

First error state will occur directly when i=0 , here j should become zero and enter the loop and calculate the addition of temp with arr[j] but its not happening since range(0,0) is not giving j any value to enter loop and execute as its meant to be.

Input: x=[4,0,-2,3]

Expected output : 5

Actual output :4

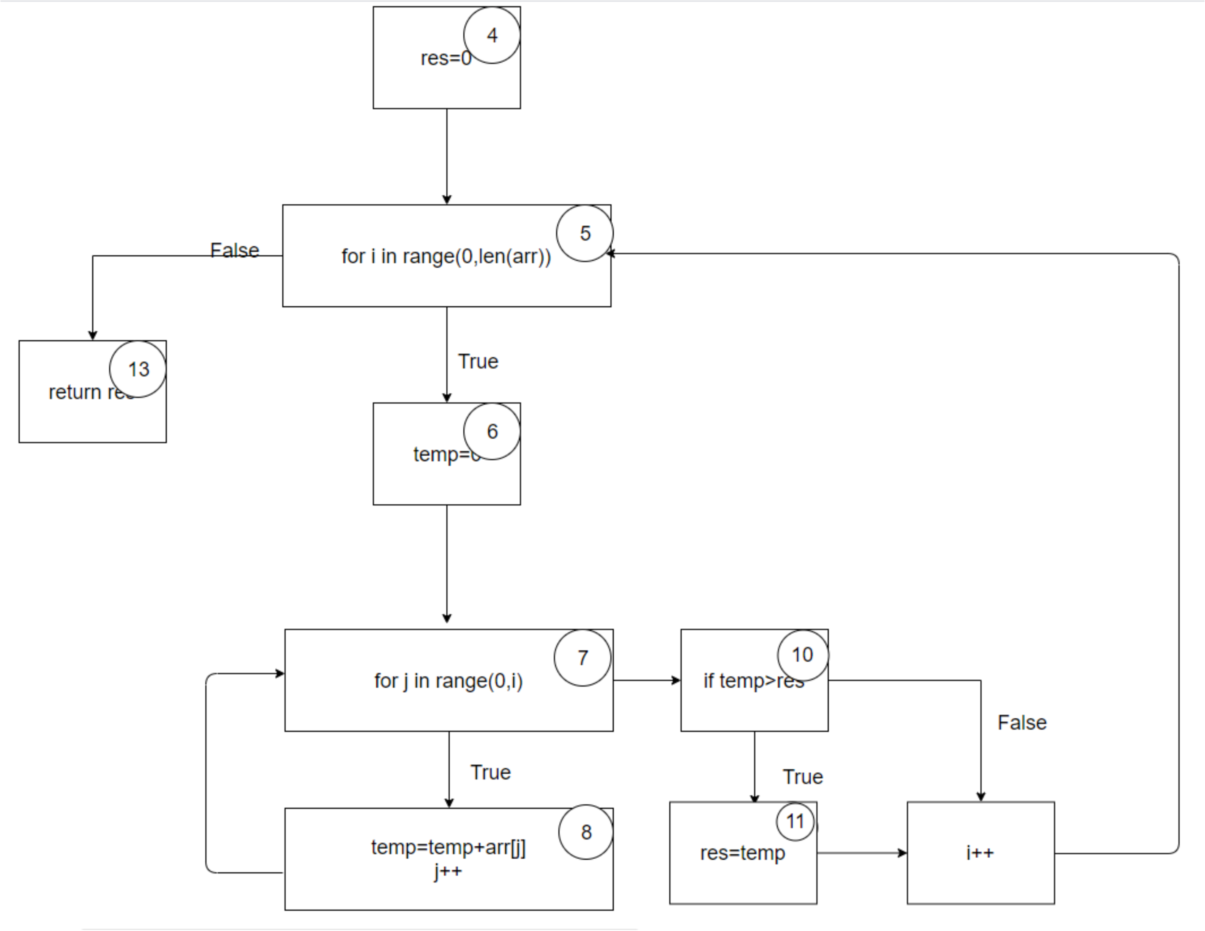
First error state:

X=[4,0,-2,3]

i =0, j= undefined

pc: if temp>res (implies if 0>0, since temp=0, res=0 at this point)

1(e):



res=0
4
for i in range(0,len(arr))
5
False
return res
13
True
temp=0
6
for j in range(0,i)
7
True
temp=temp+arr[j]
j++
8
if temp>res
10
True
False
res=temp
11
i++

2(a):

```
class RepeatUntilStmt (Stmt):
        def __init__(self, cond, statement):
                self.cond = cond
                self.statement =statement
          def __eq__ (self, other):
                return type(self) == type(other) and \
                self.cond == other.cond and \
                self.statement == other.statement
```

2(b):

$$\frac{< S, q >\Downarrow q' \quad < b, q' >\Downarrow true}{< repeat\ S\ until\ b, q >\Downarrow\ q'}$$

$$\frac{< S, q >\Downarrow q' \quad < b, q' >\Downarrow false \quad < repeat\ S\ until\ b, q' >\Downarrow\ q''}{< repeat\ S\ until\ b, q >\Downarrow\ q''}$$

2(c):

$$\frac{< x := 2, [\,] >\Downarrow [x := 2] \quad < x := x - 1, [x := 2] >\Downarrow [x := 1] \quad < x \le 0, [x := 1] >\Downarrow false \quad < x := x - 1, [x := 1] >\Downarrow [x := 0] \quad < x \le 0, [x := 0] >\Downarrow true}{< x := 2; repeat\ x := x - 1\ until\ x \le 0\ , [\,] >\Downarrow [x := 0]}$$

2(d):

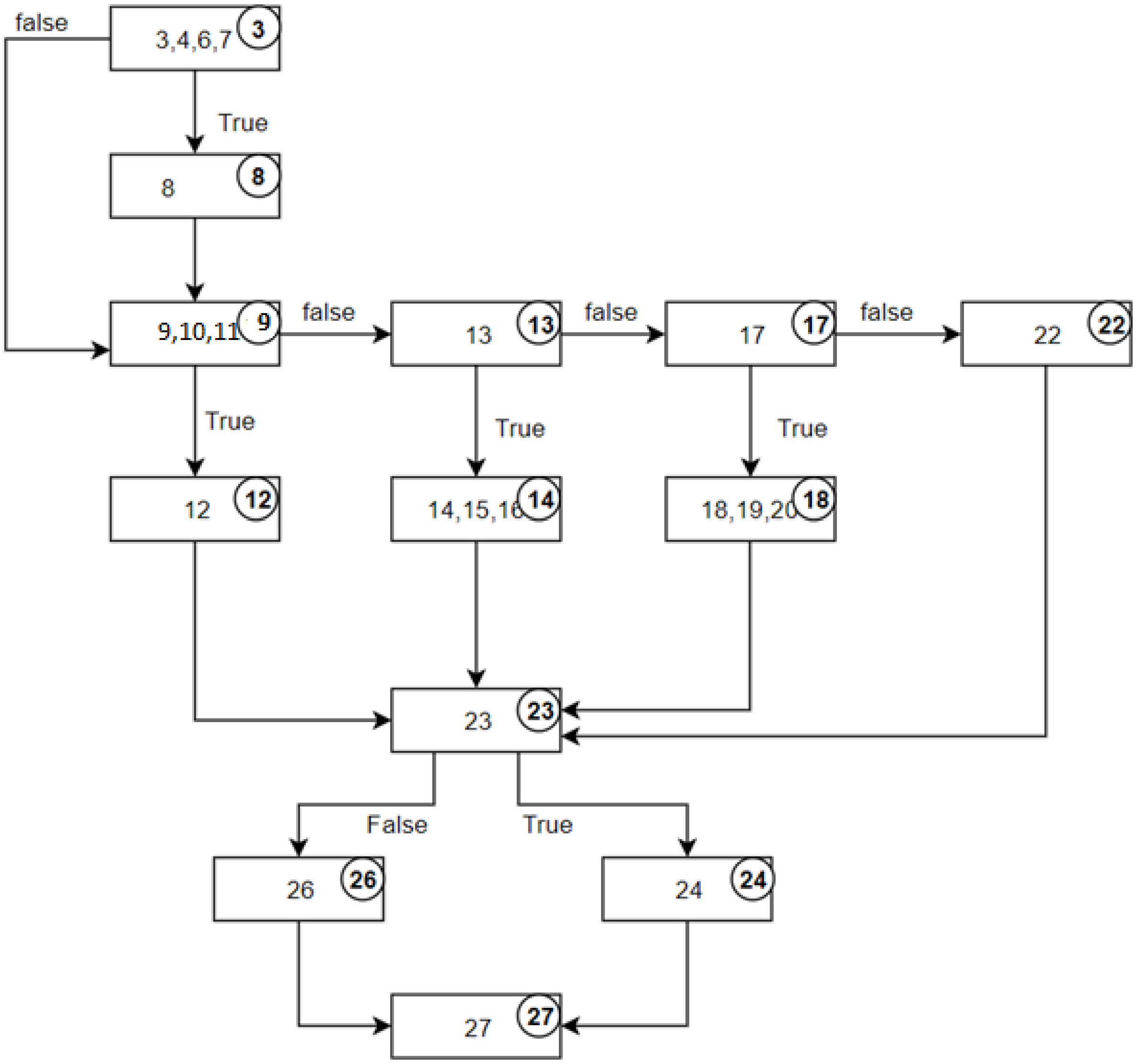
Semantics for S;while not b do S

$$\frac{< S, q >\Downarrow q' \quad < b, q' >\Downarrow true}{< S; while\ not\ b\ do\ S, q >\Downarrow q'}$$

$$\frac{< S, q >\Downarrow q' \quad < b, q' >\Downarrow false \quad < S; while\ not\ b\ do\ S, q' > q''}{< S; while\ not\ b\ do\ S, q >\Downarrow q''}$$

From observations of repeat S until b semantics from 2(b) and above we can infer clearly that both repeat S until b and S;while not b do S are resulting in the same state which implies they are semantically equivalent.

3(a):


false
3,4,6,7
3
True
8
8
9,10,11
9
false
13
13
false
17
17
false
22
22
True
True
True
12
12
14,15,16
14
18,19,20
18
23
23
False
True
26
26
24
24
27
27

3(b):

Test requirements are as follows:

Node Coverage:

TR_{NC} ={3,8,9,12,13,14,17,18,22,23,26,24,27}

Edge Coverage:

TR_{EC}={ [3,8], [3,9], [8,9], [9,12], [9,13], [13,14], [13,17], [17,18], [17,22], [12,23], [14,23], [18,23], [22,23], [23,26], [23,24], [26,27], [24,27]

Edge-Pair Coverage:

TR_{EPC}={ [3,8,9], [3,9,12], [3,9,13], [8,9,12], [8,9,13], [9,12,23], [9,13,14], [9,13,17], [13,14,23], [13,17,18], [13,17,22], [17,18,23], [17,22,23], [22,23,24], [12,23,26], [14,23,24], [18,23,24], [23,26,27], [23,24,27] }

Infeasible paths=

[22,23,26] => This path is not possible since 22 will occur only when n>0 since n cannot be negative as it's the output of length function. So at 23 it will always satisfy If condition ie., n>0 and will never go to else block ie., 26

[12,23,24]=> This path is not possible since 12 will occur only when n is 0. So at 23 it will always fail the condition n>0 and will always enter else block only.

[14,23,26]=> This path is not possible since 14 will occur when n is 1. So at 23 it will always satisfy the if condition n>0 and will never enter else block ie., 26.

[18,23,26]=> This path is not possible since 18 will occur when n is 2. So at 23 it will always satisfy the if condition n>0 and will never enter else block ie., 26.

Prime Path Coverage:

TR$_{PPC}$={ [3,8,9,12,23,26,27], [3,9,12,23,26,27], [3,8,9,13,14,23,24,27], [3,8,9,13,17,18,23,24,27], [3,8,9,13,17,22,23,24,27], [3,9,13,14,23,24,27], [3,9,13,17,18,23,24,27], [3,9,13,17,22,23,24,27] }

Infeasible paths=

[3,8,9,12,23,24,27]=> This path is not possible since 12 will occur only when n is 0. So at 23 it will always fail the condition n>0 and will always enter else block only.

[3,9,12,23,24,27]=> This path is not possible since 12 will occur only when n is 0. So at 23 it will always fail the condition n>0 and will always enter else block only.

[3,8,9,13,14,23,26,27]=> This path is not possible since 14 will occur when n is 1. So at 23 it will always satisfy the if condition n>0 and will never enter else block ie., 26.

[3,8,9,13,17,18,23,26,27]=> This path is not possible since 18 will occur when n is 2. So at 23 it will always satisfy the if condition n>0 and will never enter else block ie., 26.

[3,8,9,13,17,22,23,26,27]=> This path is not possible since 22 will occur only when n>0 since n cannot be negative as it's the output of length function. So at 23 it will always satisfy If condition ie., n>0 and will never go to else block ie., 26

[3,9,13,14,23,26,27]=> This path is not possible since 14 will occur when n is 1. So at 23 it will always satisfy the if condition n>0 and will never enter else block ie., 26

[3,9,13,17,18,23,26,27]=> This path is not possible since 18 will occur when n is 2. So at 23 it will always satisfy the if condition n>0 and will never enter else block ie., 26.

[3,9,13,17,22,23,26,27]=> This path is not possible since 22 will occur only when n>0 since n cannot be negative as it's the output of length function. So at 23 it will always satisfy If condition ie., n>0 and will never go to else block ie., 26