Assignment 2

Software Testing Methods

SS G552

in

Master of Engineering

By

Aniket Mourya (2020H1120298P)

Under the supervision of

Dr. Mukesh Kumar Rohil

Assistant Professor

Department of Computer Science and Information Systems

Birla Institute of Technology and Science

Pilani, Pilani Campus, Rajasthan

08 December 2021

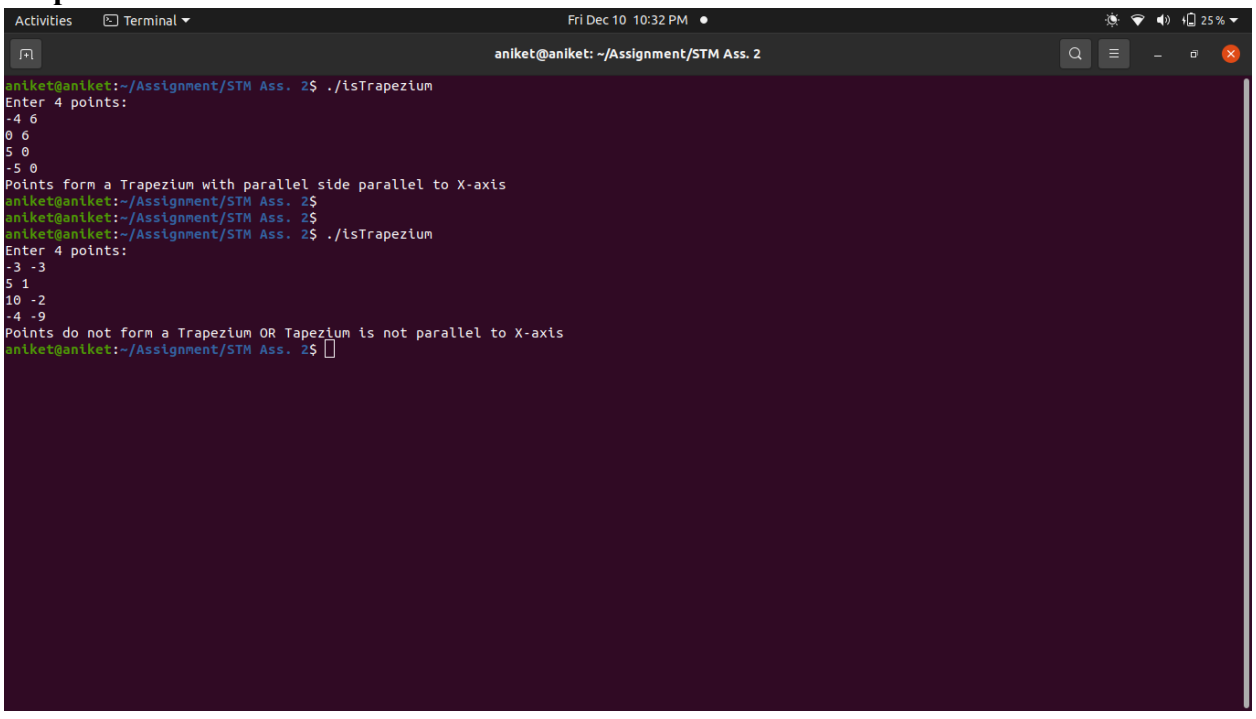
Q1. Given four points in 2D check if they form a trapezium with parallel sides parallel to X-axis. If not, simply state that it is not the case.

Github link: https://github.com/aniketmourya/Trapezium.git

```
// Language Used: C++
#include<bits/stdc++.h>
using namespace std;
// structure to create a point
struct Point{
// x, y coordinates of a point
 double x;
 double y;
// constructor to store a point
    Point(double x, double y) {
        this->x = x;
        this->y = y;
    }
};

bool isTrapezium(struct Point P1, struct Point P2, struct Point P3, struct Point P4){
        // finding slope of line formed by Points 1 & 2
        double slope12 = (P1.y - P2.y) / (P1.x - P2.x);

        // finding slope of line formed by Points 3 & 4
        double slope34 = (-P3.y + P4.y) / (-P3.x + P4.x);

        // finding slope of line formed by Points 1 & 4
        double slope14 = (P1.y - P4.y) / (P1.x - P4.x) ;

        // finding slope of line formed by Points 2 & 3
        double slope23 = (P2.y - P3.y) / (P2.x - P3.x) ;

        // two sides are parallel if the corresponding slope of sides are same
        /*Finding and checking if slope of line  formed by P1 and P2 is same as
          that of line formed by P3 and P4 and if parallel sides are parallel to X-axis.*/

        /* Finding and checking if slope of line  formed by P1 and P4 is same as
          that of line formed by P2 and P3 and if parallel sides are parallel to X-axis.*/

        if((slope12==slope34) && ((P1.y-P2.y)==0) || ((slope14==slope23) && ((-P3.y + P4.y)
== 0))){
                return true;
        }
        else{                       // if it not a trapezium  OR parallel side not Parallel to X-axis
                return false;
```

```
        }
}

int main(){
        double x,y;
        cout<<"Enter 4 points: "<<endl;
        cin>>x>>y;
        Point P1(x,y);
        cin>>x>>y;
        Point P2(x,y);
        cin>>x>>y;
        Point P3(x,y);
        cin>>x>>y;
        Point P4(x,y);
        bool checkTrapezium =  isTrapezium(P1, P2, P3, P4); // check if points form a trapezium
                if(checkTrapezium){
                   cout<<"Points form a Trapezium with parallel side parallel to
X-axis"<<endl;
                }
                else{
                   cout<<"Points do not form a Trapezium OR Tapezium parallel side is not
parallel to X-axis"<<endl;
                }
        return 0;
}
```

Output:



Q2. Draw the control flow Graph of the program for the program developed above.

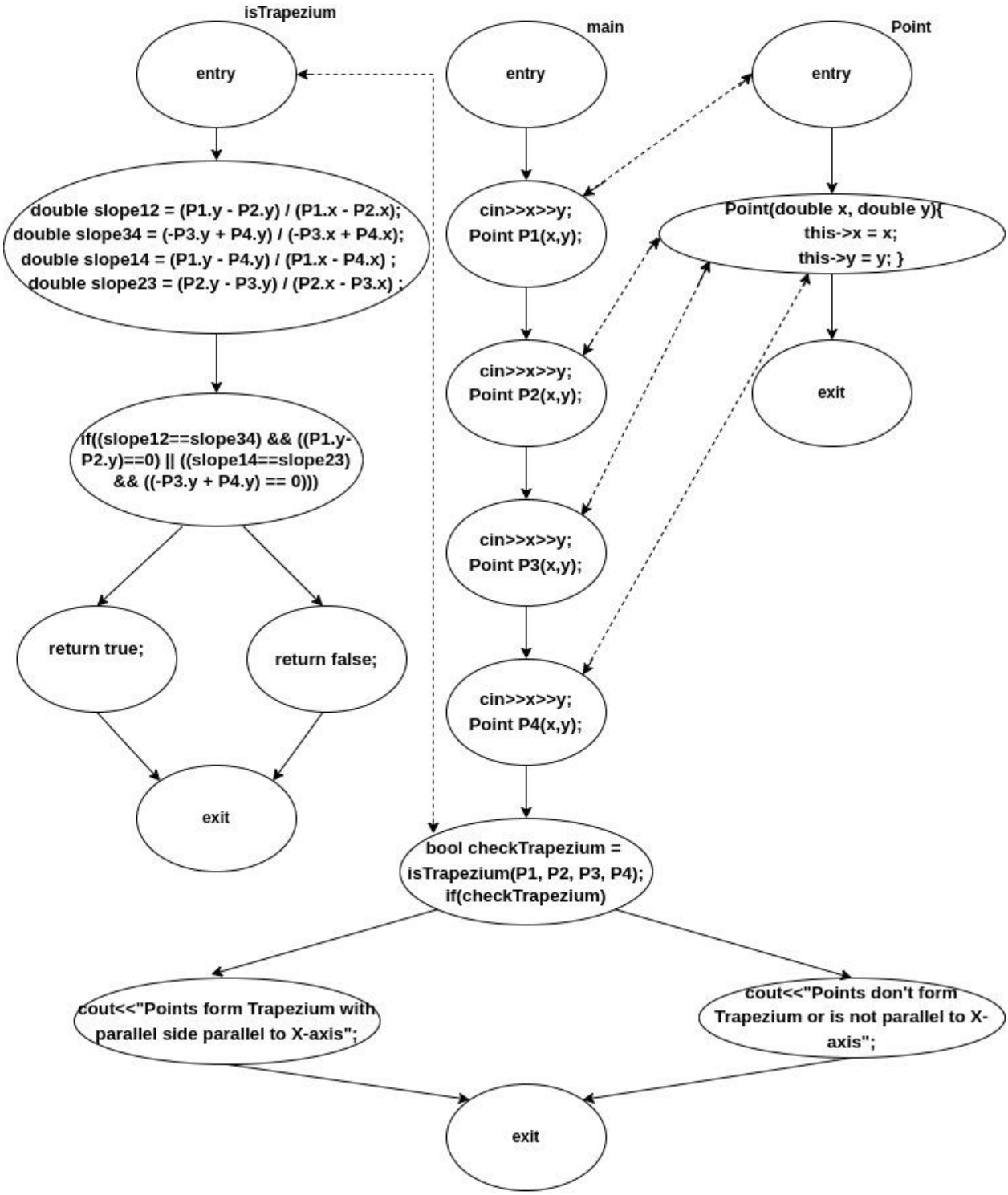


Fig. CFG to check if they form a trapezium with parallel sides parallel to X-axis

Q3. Derive minimal MC/DC tests for the program developed above.

Condition A:

((slope12==slope34) && ((P1.y-P2.y)==0) || ((slope14==slope23) && ((-P3.y + P4.y) == 0))

C1 = slope12 == slope34
C2 = (P1.y - P2.y) == 0
C3 = slope14 == slope23
C4= P4.y - P4.y

Test case number	C1	C2	C3	C4	(C1 && C2) || (C3 && C4)
1	T	T	T	F	T
2	T	F	T	T	T
3	T	F	F	T	F
4	T	F	T	F	F
5	F	T	T	F	F

Therefore minimal MC/DC test cases for the above condition is 1, 2, 3, 4, 5

Condition B:

C1 = checkTrapezium == 1

Test case number	C1	Output
1	T	T
2	F	F

Therefore minimal MC/DC test cases for the above condition is 1 and 2

4. Draw a data flow graph for the program developed above.

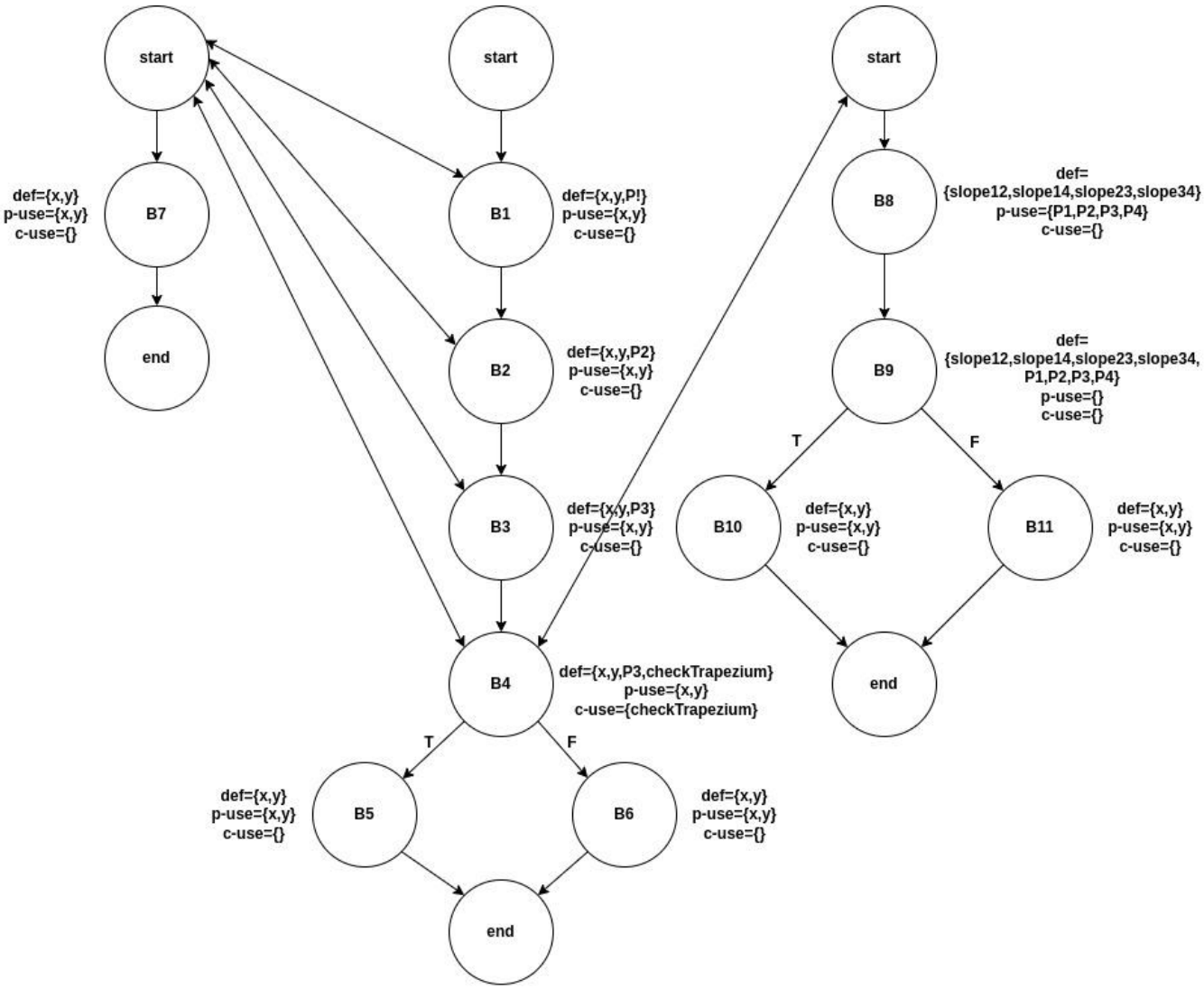


Fig. Data flow graph to check if they form a trapezium with parallel sides parallel to X-axis

5. Derive All-uses coverage tests for the program developed above.

Variable	All-uses coverage test
x,y	[Start,1,2,3,4,5,end], [Start,1,2,3,4,6,end]
checkTrapezium	[Start,1,2,3,4,5,end], [Start,1,2,3,4,6,end]
P1(formal arguments)	[Start,1,2,3,4,start,8,9,10,end] [Start,1,2,3,4,start,8,9,11,end]
P2(formal arguments)	[Start,1,2,3,4,start,8,9,10,end] [Start,1,2,3,4,start,8,9,11,end]
P3(formal arguments)	[Start,1,2,3,4,start,8,9,10,end] [Start,1,2,3,4,start,8,9,11,end]
P4(formal arguments)	[Start,1,2,3,4,start,8,9,10,end] [Start,1,2,3,4,start,8,9,11,end]
slope12	[Start,1,2,3,4,start,8,9,10,end] [Start,1,2,3,4,start,8,9,11,end]
slope34	[Start,1,2,3,4,start,8,9,10,end] [Start,1,2,3,4,start,8,9,11,end]
slope14	[Start,1,2,3,4,start,8,9,10,end] [Start,1,2,3,4,start,8,9,11,end]
slope23	[Start,1,2,3,4,start,8,9,10,end] [Start,1,2,3,4,start,8,9,11,end]
P1(actual arguments)	[Start,1,2,3,4,5,end], [Start,1,2,3,4,6,end]
P2(actual argumentsr)	[Start,1,2,3,4,5,end], [Start,1,2,3,4,6,end]
P3(actual arguments)	[Start,1,2,3,4,5,end], [Start,1,2,3,4,6,end]
P4(actual arguments)	[Start,1,2,3,4,5,end], [Start,1,2,3,4,6,end]

Table A: All usage coverage tests