Home Work 11

The problems in this problem set cover lectures C13 and C14

1.

a. Define a robust algorithm to carry out integer division using repeated subtraction. Your algorithm accepts two integers and returns the quotient and the remainder. Hint: What are the preconditions and postconditions of your algorithm?

Precondtions: Two integers *x*,*y*
y is non-zero

Algorithm:

```
Set R to absolute_value(x)
Set Q to zero
While R >= absolute_value (y)
        Increment Q
        R := R- absolute_value(y)
If either x or y are negative
        If both x and y are negative
                Set R to –R
        else
                if x is negative
                        Set R to –R
                Set Q to –Q
Display Q and R
```

Postconditions: Q contains the quotient
R contains the remainder
$x = Q*y + R$, abs(R) < abs(Q)

b. Implement your algorithm as an Ada95 program, using exception handling to provide robustness.

```
1. ------------------------------------------------------
2. -- Procedure to carry out robust division
3. -- Programmer: Jayakanth Srinivasan
4. -- Date Last Modified : April 17,2004
5. ------------------------------------------------------
6.
7. with Ada.Text_Io;
8. with Ada.Integer_Text_Io;
9. use Ada.Text_Io;
10. use Ada.Integer_Text_Io;
11.
12. procedure Robust_Division is
13.    X,
14.    Y,
15.    Q,
16.    R             : Integer;
17.    Divide_By_Zero : exception;
18.
19. begin
20.    loop
21.       Ada.Text_IO.Skip_Line;
22.       begin
23.          -- get the dividend (X)
24.          Ada.Text_Io.Put("Please Enter the X : ");
25.          Ada.Integer_Text_Io.Get(X);
26.          Ada.Text_Io.Skip_Line;
27.
28.          -- get the divisor (Y)
29.          Ada.Text_Io.Put("Please Enter the Y : ");
30.          Ada.Integer_Text_Io.Get(Y);
31.          Ada.Text_Io.Skip_Line;
32.
33.          if Y = 0 then
34.             raise Divide_By_Zero;
35.          end if;
36.
37.          --set the remainder to absolute value of X
38.          R :=abs(X);
39.          -- set quotient to zero
40.          Q := 0 ;
41.          -- while remainder is greater than absolute value of y
42.          while  R >= abs(Y)  loop
43.             -- deduct absolute value of y from the remainder
44.             R := R - abs(Y) ;
45.             -- increment the quotient
46.             Q := Q + 1;
47.          end loop;
48.
49.          --ensure that the sign on the quotient is quotient
50.          if (X<0) or (Y<0) then
51.             if (X<0) and (Y<0) then
52.                -- if both x,y are negative then remainder is negative
53.                R := -1*R;
```

```
54.            else
55.              if (X<0) then
56.                -- if X is negative then remainder is negative
57.                R:= -1*R;
58.              end if;
59.              -- if either x or y not both, then quotient is negative
60.              Q := -1*Q;
61.            end if;
62.         end if;
63.         -- Display the quotient
64.         Ada.Text_Io.Put_Line(Integer'Image(Q));
65.
66.         -- display the remainder
67.         Ada.Text_Io.Put_Line(Integer'Image(R));
68.
69.         -- if the program has reached this part, there were no exceptions
70.         exit;
71.
72.
73.       exception
74.         when Data_Error =>
75.           Ada.Text_Io.Put_Line("Trying to enter a non-integer");
76.
77.         when Divide_By_Zero =>
78.           Ada.Text_Io.Put_Line("Trying to divide by zero");
79.
80.         when others =>
81.           Ada.Text_Io.Put_Line("Dont know what this exception is");
82.
83.           -- this is the end of the block created by the begin statement
84.       end;
85.       -- this is the end of the loop
86.     end loop;
87.
88. end Robust_Division;
```

88 lines: No errors

2.

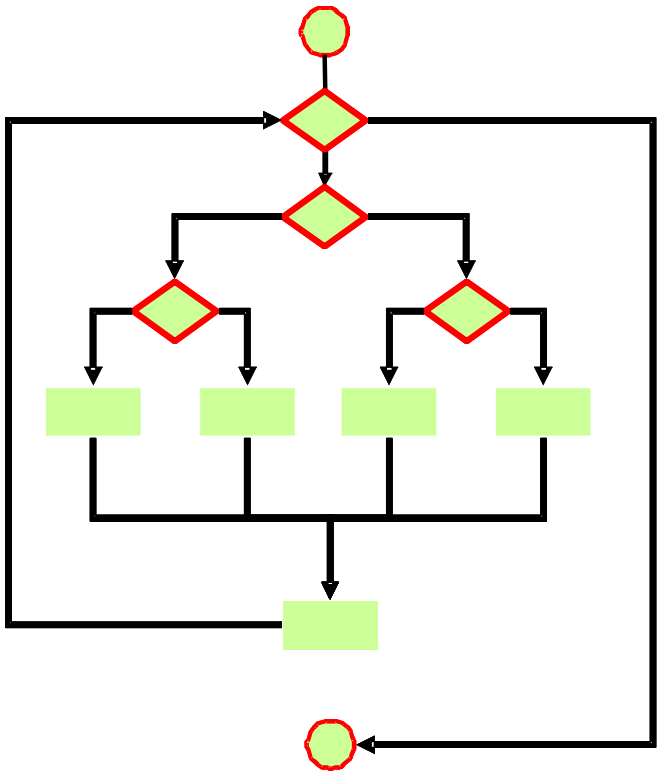
a. What is the cyclomatic complexity of the code fragment shown below?

```
loop
  exit when Flag := True;

  if A < 100 and B > 200 then
    if A > 50 then
      Sum := Sum +2;
    else
      Sum := Sum +1;
    end if;
  else
    if B < 300 then
      Sum:= Sum -1;
    else
      Sum := Sum -2;
    end if;
  end if;

end loop;
```

Hint: Draw the control flow graph



11 Nodes, 14 edges => Cyclomatic complexity = 5.

b. What is the minimum number of test cases needed to test the fragment of code shown below? Justify your answer.

```
1. if A < 100 and B > 200 then
2.       if A > 50 then
3.          Sum := Sum +2;
4.       else
5.          Sum := Sum +1;
6.       end if;
7.    else
8.       if B < 300 then
9.          Sum:= Sum -1;
10.      else
11.           Sum := Sum -2;
12.      end if;
13. end if;
```

Test Case	A	B	Line Tested
1	50 < A < 100	B > 200	Sum:=Sum+2
2	A <= 50	B > 200	Sum:=Sum+1
3	A >=100	B < 300	Sum:=Sum-1
4	Any Other combination of A and B		Sum:=Sum-2