C11 Solutions

1.

```
Count := 1;
FOR I in 1 .. 10 LOOP
        If I MOD 2 = 0 THEN
                FOR J in 1 .. 10 LOOP
                        Count:= Count + 2;
                END LOOP;
        ELSE
                FOR J in 1 .. 5 LOOP
                        Count := Count – 1;
                END LOOP;
        END IF;
END LOOP;
```

Count = 76.

Count increments by 20 when I is even and decrements by 5 when I is odd.

2. Write an Ada95 program to implement the Euler's 2nd order integration method? Turn in a hard copy of your algorithm and code listing and an electronic copy of your code.

C 11 part b ALGORTIHM

Eluer's 2nd order integration – use trapeziodal rule.
Area of a trapezoid under curve = .5*(y1+y2)*delta_x

Algortihm:
Ask user for inputs:

- Coefficients of each polynomial term plus constant
- Upper and Lower Bounds of integration
- Step Size

Calculate number of steps = (upper_bound-lower_bound)/step_size and convert it to an integer

Loop from 0 to the number of steps using a for loop, performing euler's second order approximation

- Integral = Integral + .5*(y1+y2)*step_size
- Y1 = Y2
- Y2 = Y2 +Step_Size

Print out results

C11 b code
Solutions

```
with Ada.Text_Io;
use Ada.Text_Io;
with Ada.Integer_Text_Io;
use Ada.Integer_Text_Io;
with Ada.Float_Text_Io;
use Ada.Float_Text_Io;

procedure Second_Order_Euler is
   --procedure to perform euler's second oreder integration method
   --a definite integral is input by the user and is the calculation is
   --performed and returned
   --only takes in polynomials up to 6th order
   --Unified Computers and Programming, Problem C11 b, Fall 2003
   --Author: Howard Kleinwaks, based on an algorithm by Phil Springmann
   --Last Modified: October 5, 2003

   --declare variables
   Order : Integer; --stores order of polynomial
   Upper_Bound : Float;
   Lower_Bound : Float; --numbers to store the bounds of the integral
   First_Order_Term : Float;
   Second_Order_Term : Float;
   Third_Order_Term : Float;
   Fourth_Order_Term : Float;
   Fifth_Order_Term : Float;
   Sixth_Order_Term : Float;
   Constant_Term : Float;
   Integral : Float;
   Step_Size : Float; --input value by user to determine step size to use
   Number_Of_Steps : Float;
   Integer_Number_Of_Steps : Integer; --need integer number of steps to
use in for loop
   Low_Step : Float;
   High_Step : Float; --variables to represent current x-values (x_i and
 x_i+1)

begin -- Second_Order_Euler
   --get input variables
   --take the order of the polynomial and the coefficients
   Put("Please enter the order of the polynomial (between one and six):"
);
   Get(Item => Order);
   New_Line;
   --check order to make sure it is within the proper bounds
   while Order< 1 and Order > 6 loop
      Put("Please enter the order of the polynomial (between one and six
):");
      Get(Item => Order);
      New_Line;
   end loop;

   --get coefficients of the polynomial
   Put("Please enter the constant term:");
   Get(Item => Constant_Term);
   New_Line;
   Put("Please enter the coefficent of the lowest order term:");
   Get(Item => First_Order_Term);
   New_Line;
   Put("Please enter the coefficent of the next lowest order term:");
   Get(Item => Second_Order_Term);
```

```
New_Line;
Put("Please enter the coefficent of the next lowest order term:");
Get(Item => Third_Order_Term);
New_Line;
Put("Please enter the coefficent of the next lowest order term:");
Get(Item => Fourth_Order_Term);
New_Line;
Put("Please enter the coefficent of the next lowest order term:");
Get(Item => Fifth_Order_Term);
New_Line;
Put("Please enter the coefficent of the next lowest order term:");
Get(Item => Sixth_Order_Term);
New_Line;

--get bounds of integration
Put("Please enter the lower bound:");
Get(Item => Lower_Bound);
New_Line;
   Put("Please enter the upper bound:");
   Get(Item => Upper_Bound);
   New_Line;

   --get step size desired from user
   Put("Please enter the step size:");
   Get(Item => Step_Size);
   New_Line;

   --calculate number of steps
   Number_Of_Steps := (Upper_Bound - Lower_Bound)/Step_Size;
   --convert to integer
   Integer_Number_Of_Steps := Integer(Number_Of_Steps);

   --now loop from 0 to the number of steps, performing euler's second o
rder approximation
   --the approximation follows the trapezoidal rule
   --area of a trapezoid = .5*(b1 + b2)*h, where b1 and b2 are the funct
ion values at either end of the step
   --and h is the step size

   --need to initialize the value of integral (the result) and Low_Step
and High_Step
   Integral := 0.0;
   Low_Step := Lower_Bound;
   High_Step := Lower_Bound + Step_Size;

   for I in 1..Integer_Number_Of_Steps loop
      --calculate integral according to following method:
      --Integral := Integral + .5*(f(x)+f(x+1))*Step_Size
      Integral := Integral + 0.5*((Sixth_Order_Term*High_step**6 + Fifth
_Order_Term*Low_Step**5 + Fourth_Order_Term*Low_Step**4
            + Third_Order_Term*Low_Step**3 + Second_Order_Term*Low_Step*
*2 + First_Order_Term*Low_Step+Constant_Term)
         + (Sixth_Order_Term*High_Step**6 + Fifth_Order_Term*High_Step**
5 + Fourth_Order_Term*High_Step**4
            + Third_Order_Term*High_Step**3 + Second_Order_Term*High_Ste
p**2 + First_Order_Term*High_Step+Constant_Term))*Step_Size;
      Low_Step := Low_Step+Step_Size;
      High_Step := High_Step+Step_Size;
   end loop;
   Put("The integration is: ");
```

```
   Put(Integral, Exp => 0);
end Second_Order_Euler;
```

3.

Algorithm:

1. Initialize the counter to 1
2. Initialize Sum to 0
3. While (counter <= 10) loop
    i. Get a number from the user
    ii. Add Number to Sum
    iii. Increment the Counter
4. Compute the average by dividing sum by 10
5. Display computed average to the user

Code Listing

GNAT 3.13p (20000509) Copyright 1992-2000 Free Software Foundation, Inc.

Compiling: c:/docume~2/joeb/desktop/16070/codeso~1/average_with_while.adb (source file time stamp: 2003-10-02 02:41:10)

```
 1. ---------------------------------------------------
 2. -- Program to find the average of 10 numbers using
 3. -- a While Loop
 4. -- Programmer : Joe B
 5. -- Date Last Modified : October 01, 2003
 6. ---------------------------------------------------
 7.
 8.
 9. with Ada.Text_Io;
10. with Ada.Float_Text_Io;
11.
12. procedure Average_With_While is
13.    Counter : Integer :=1; -- initialize counter to 0
14.    Sum : Float :=0.0; -- initializise sum to 0
15.    Num : Float;-- variable used to get input from the user
16. begin
17.    while (Counter <= 10) loop
18.       -- get input from the user
19.       Ada.Text_Io.Put("Please Enter A Number : ");
20.       Ada.Float_Text_Io.Get(Num);
21.       Ada.Text_Io.Skip_Line;
22.       -- compute sum
23.       Sum := Sum + Num;
24.       -- increment the counter
25.       Counter := Counter +1;
26.
27.    end loop;
28.
29.    Ada.Text_Io.Put("The Average of Numbers is :");
30.    Ada.Float_Text_Io.Put(Sum/10.0);
31.
32. end Average_With_While;
```

32 lines: No errors