16.unified
Introduction to Computer Programming

Examination
11/18/05

Professor I. Kristina Lundqvist,
Lecturers Heidi Perry and Gustaf Naeser
Fall 2005

You have 55 minutes to take this examination. Do not begin until you are instructed to do so. This is a closed book examination. No external materials are permitted, including calculators or other electronic devices. All answers must be written in the examination paper. This examination should consist of **11** pages (including this cover sheet). Count the number of pages and immediately report any discrepancy. Should you need to do so, you may continue your answer on the back pages. Put your name at the bottom of each page of this exam.

Question 1 True/False (10)	
Question 2 Read Ada Code (20)	
Question 3: Recursion (10)	
Question 4: Arrays (15)	
Question 5: Number Conversions (15)	
Question 6: Computer Architecture (10)	
Question 7: Rover (20)	

Name_____

Problem 1

True or False (10p)
(1p each)

For each statement, indicate if it is true (T) or false (F).

The strong typing of Ada95 prevents users from confusing variables of different types.	
Iteration and recursion solves the same kind of problem: repeated execution of statements.	
A string in Ada95 can be treated as records of characters.	
Variable declarations implicitly set default values of the declared variable.	
Variables used as actual parameters must have the same name as the name of the subprogram's formal parameters.	
All kinds of loops can be described using for loops.	
A function that returns two values must be implemented as a procedure.	
Distinct datatypes, e.g. type My_Type is new Integer, are for creating types that should not be confused with other types using the same symbols (possible values).	
Subtypes are assignment compatible with the base type.	
You can store both positive and negative numbers using 2-complement	

Problem 2   *Reading Ada Code (20p)*

Our programmer friend Joe D has written the program `main.adb`. Predict what the output of Joe's program will be at commented lines A through F.

`main.adb`

```ada
with Ada.Integer_Text_Io;
use Ada.Integer_Text_Io;

procedure Main is

   A : Integer := 2;
   B : Integer := 5;
   C : Integer := 4;

   function F1 (X: Integer;Y :Integer) return Integer is
      C : Integer;
   begin
      C := X+Y;
      return C;
   end;

   procedure F2 (X: in out Integer; Y : in Integer) is
   begin
      A := X;
      X := A*Y;
   end;

begin
   B:=A*B;
   A:=F1(B,C);
   Put(A);    -- Part A

   B:=B+C;
   Put(B);    -- Part B

   A:=A-C;
   F2(B,C);
   Put(B);    -- Part C
   Put(A);    -- Part D

   C:=F1(A,B);
   Put(C );    -- Part E

   for C in 1..A loop
      B:=B+1;
   end loop;
   Put(B);    -- Part F
   Put(C);    -- Part G

end Main;
```

Part A =   Part D =   Part G =

Part B =   Part E =

Part C =   Part F =

Problem 3   *Recursion (10p)*

What is wrong with the following recursive function and how should it be corrected?
(Hint, look at the results from the two calls to it.)

Indicate the change that should be done and give a short motivation/explanation of no
more than two lines.

recursiveaddition.adb

```ada
with Ada.Integer_Text_IO;
use Ada.Integer_Text_IO;

procedure RecursiveAddition is

   function Add_Every_Other_From_Zero(N : Integer) return Integer is
   begin
      if (N = 0) then
         return 0;
      else
         return N + Add_Every_Other_From_Zero(N-2);
      end if;
   end Add_Every_Other_From_Zero;

begin
   Put(Add_Every_Other_From_Zero(8));
   Put(Add_Every_Other_From_Zero(7));
end RecursiveAddition;
```

Problem 4 *Arrays and Records (15p)*

a. Consider the program `arrayinit.adb`.

`arrayinit.adb`
```
with Ada.Integer_Text_IO;
use Ada.Integer_Text_IO;

procedure ArrayInit is
   type TenArray is array (1..10) of Integer;
   Chunk : TenArray;
begin
   for I in TenArray'Range loop
      Put(Chunk(I));
   end loop;
end ArrayInit;
```

What warnings or errors can be expected during compilation of the program? (3p)

What will the output be when running the program? (2p)

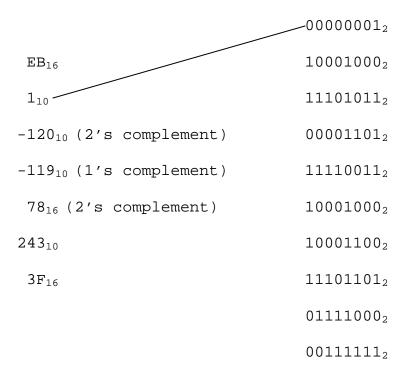Write code to correct the problem and indicate where the code should go. (5p)

b. Consider the program in `nutsandbolts.adb`. (5 p)

nutsandbolts.adb

```ada
with Ada.Text_Io, Ada.Integer_Text_Io;
use Ada.Text_Io, Ada.Integer_Text_Io;

procedure NutsAndBolts is

   type Description is
      record
         L    : Integer;
         H    : Integer;
         W    : Integer;
         Cost : Float;
      end record;

   type Sets is array (1..30) of Description;
   type Part is (Bolt, Nut, Cap);
   type Catalog is array (Part'range) of Sets;

   type Warehouse is
   record
      Bolts_R_Us: Catalog;
      Hold_Tighter : Catalog;
   end record;

   Dist_Center : Warehouse;

begin
   -- Set length (L) of the 10th Nut in the Bolts_R_Us Catalog to 10 HERE.


end NutsAndBolts;
```

How would you go about setting the length (L) of the 10th Nut in the Bolts_R_Us Catalog to 10? Write the statement for the assignment in the code above. (10p)

Problem 5   *Number Conversions (15 P)*

a. Match the decimal and hexadecimal numeric values on the left to their equivalent binary numeric values on the right. (12p)

00000001_2

EB_{16} 10001000_2

1_{10} 11101011_2

-120_{10} (2's complement) 00001101_2

-119_{10} (1's complement) 11110011_2

78_{16} (2's complement) 10001000_2

243_{10} 10001100_2

$3F_{16}$ 11101101_2

01111000_2

00111111_2

b. Convert the following floating point number to decimal notation. Note: The number has been stored using scientific notation, i.e., the leftmost 1 has been removed. (3p)

111011001100 with 1 signbit, 4 bits exponent (excess4) and 7 bits mantissa =

Assume you have an 8-bit computer that contains two registers (R1 and R2) to contain data the CPU will use for arithmetic operations. This computer is designed to process the following legal Opcodes:

000 = stop
001 = load data stored at indicated address into R1
010 = load data stored at indicated address into R2
011 = store R1 into address location given
100 = store R2 into address location given
101 = add R1 to R2 (results stored in R2)

The three most significant bits contain the op-code. The remaining 5 bits contain the address of any data that is required by the operation.

a. The machine level program that is written below is equivalent to the Ada statement of Y:=X + Y + Z; where X, Y, and Z are stored in addresses 10_{10}, 11_{10} and 12_{10} respectively. Fill in the meaning of each of the statements. The first one has already been done for you.

OpCode	Address	Meaning
010	01011 (11_{10})	Load data stored at address 11_{10} (the # Y) into R2
001	01010 (10_{10})	
101	00000	
001	01100 (12_{10})	
101	00000	
100	01011	
000	00000	

b. Given the contents of memory BEFORE the execution of the program, fill in the contents of memory AFTER the above machine instruction program has executed:

Address	Data (memory contents)
00001010 (10_{10})	00000011
00001011 (11_{10})	00001001
00001100 (12_{10})	00010000

BEFORE

Address	Data (memory contents)
00001010 (10_{10})	
00001011 (11_{10})	
00001100 (12_{10})	

AFTER

Problem 7   *The Rover (20 P)*

Student Ida Know has written the following lego program for the Mars Rover you built for this class. She did not comment it well at all. And some of the procedure names are boring and do not reflect the actual purpose of the code.

a. Comment her code in the areas that are outlined. Describe *why* a particular section of code exists rather than just describing what it does. (6p)

```
with Lego;
use Lego;

procedure Q4 is

   Left_Wheel  : constant Output_Port := Output_A;
   Right_Wheel : constant Output_Port := Output_C;
   Left_Rot    : constant Sensor_Port := Sensor_1;
   Right_Rot   : constant Sensor_Port := Sensor_3;
   Light       : constant Sensor_Port := Sensor_2;

   procedure P1 is
   begin
      --
      Output_Power(
         Output => Left_Wheel,
         Power  => Power_High);
      Output_Power(
         Output => Right_Wheel,
         Power  => Power_High);
      --
      Config_Sensor(
         Sensor => Left_Rot,
         Config => Config_Rotation);
      Config_Sensor(
         Sensor => Right_Rot,
         Config => Config_Rotation);
      --
      Config_Sensor(
         Sensor => Light,
         Config => Config_Light);

      Output_Power(Left_Wheel,7);
      Output_Power(Right_Wheel,7);
      Clear_Sensor(Left_Rot);
      Clear_Sensor(Right_Rot);

   end P1;

   procedure Drive_Forward(Clicks:Integer) is
   begin

      -- Clear sensors and apply power to rover wheels
      Clear_Sensor(Left_Rot);
      Clear_Sensor(Right_Rot);
      Output_On_Reverse(Left_Wheel);
```

```
        Output_On_Reverse(Right_Wheel);

        while (abs(Get_Sensor_Value(Left_Rot))<Clicks) loop
            Wait(10);
        end loop;

        Output_Off(Left_Wheel);
        Output_Off(Right_Wheel);

    end Drive_Forward;

    procedure P2 is
    begin                          _____
        --
        Output_On_Forward(Left_Wheel);
        Output_On_Reverse(Right_Wheel);

        Wait(1000);
        --                         _____
        Output_Off(Left_Wheel);
        Output_Off(Right_Wheel);


    end P2;

    Value : Integer;

begin          _____
    --
    P1;
    Select_Display(Display_Sensor_2);
    loop                                        In this comment, identify what
        Drive_Forward(1000);                    the rover will do when this
                                                program executes
        Wait(100);
        Value := Get_Sensor_Value(Light);

        P2;

        Wait(100);
        Value := Get_Sensor_Value(Light);
    end loop;
end;
```

b. Rename the procedures (P1, P2) so they have more appropriate names that reflect
   what the procedure does for the lego rover (4 p)

P1_____

P2_____

c. I enjoyed the Lego portion of Unified C&P (please answer honestly). (2p)
   A. I agree
   B. I disagree
   C. Choose me and you get a zero
   D. I don't know/I don't understand.

d. Short answer: Why should you call the lego procedure Clear_Sensor for the rotation sensor before doing a new rover maneuver? (2p)

e. The Lego Rover ….(circle all that apply) (3p)

   1) runs Mission Critical Software.
   2) software is real time.
   3) is an embedded system.
   4) is an example of a Von Neuman Architecture.
   5) is used on the Pathfinder mission.
   6) is certified by NASA.
   7) has a separate floating point processor.

f. List 3 features of the standard Ada95 language that you cannot use with Ada Mindstorms and the Lego Rover (3p)

   1._____

   2._____

   3._____