Name: Luka Aitken

Student ID: T00663672

Date: February 12, 2025

SENG 4630 – Lab 6 – Programming in Ada 2012 Packages

Task 1:

```
-- t1_luka.adb
package body T1_Luka is
   procedure Min_Max(A: Int_Array; Min, Max: out Integer) is
   begin
      Min := A(A'First);
      Max := A(A'First);

      for X of A loop
         if X < Min then
            Min := X;
         elsif X > Max then
            Max := X;
         end if;
      end loop;
   end Min_Max;

   function Min_Value(A: Int_Array) return Integer is
      Min, Max: Integer;
   begin
      Min_Max(A, Min, Max);
      return Min;
   end Min_Value;

   function Max_Value(A: Int_Array) return Integer is
      Min, Max: Integer;
   begin
      Min_Max(A, Min, Max);
      return Max;
   end Max_Value;
end T1_Luka;
```

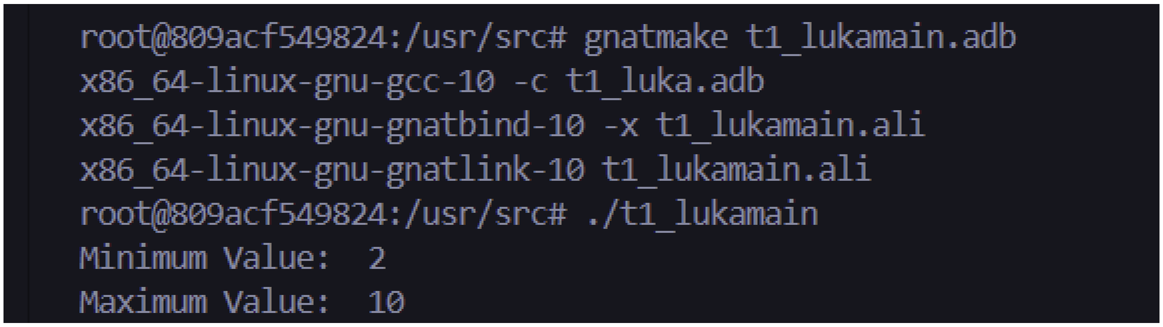
```
--t1_luka.ads
package t1_luka is
   type Int_Array is array (Integer range <>) of Integer;

   function Min_Value(A: Int_Array) return Integer;
   function Max_Value(A: Int_Array) return Integer;
end t1_luka;

--t1_lukamain.adb
with Ada.Text_IO;
with t1_luka;

use Ada.Text_IO;
use t1_luka;

procedure t1_lukamain is
   A : Int_Array(1 .. 6) := (3, 2, 7, 4, 9, 10);
begin
   Put_Line("Minimum Value: " & Integer'Image(Min_Value(A)));
   Put_Line("Maximum Value: " & Integer'Image(Max_Value(A)));
end t1_lukamain;
```



```
root@809acf549824:/usr/src# gnatmake t1_lukamain.adb
x86_64-linux-gnu-gcc-10 -c t1_luka.adb
x86_64-linux-gnu-gnatbind-10 -x t1_lukamain.ali
x86_64-linux-gnu-gnatlink-10 t1_lukamain.ali
root@809acf549824:/usr/src# ./t1_lukamain
Minimum Value:  2
Maximum Value:  10
```

Task 2:

```
--t2_luka.adb
package body t2_luka is
  procedure Selection_Sort (A : in out Int_Array) is
    Temp, Min_Index : Integer;
  begin
    for I in A'First .. A'Last - 1 loop
      Min_Index := I;
      for J in I + 1 .. A'Last loop
        if A (J) < A (Min_Index) then
          Min_Index := J;
        end if;
      end loop;
      Temp  := A (I);
      A (I)  := A (Min_Index);
      A (Min_Index) := Temp;
    end loop;
  end Selection_Sort;

  procedure Insertion_Sort (A : in out Int_Array) is
    Key, J : Integer;
  begin
    for I in A'First + 1 .. A'Last loop
      Key := A (I);
      J   := I - 1;

      while J >= A'First and then A (J) > Key loop
        A (J + 1) := A (J);
        J  := J - 1;
      end loop;

      A (J + 1) := Key;
    end loop;
  end Insertion_Sort;

  procedure Bubble_Sort (A : in out Int_Array) is
    Temp : Integer;
  begin
    for I in A'First .. A'Last loop
      for J in A'First .. A'Last - 1 loop
        if A (J) > A (J + 1) then
          Temp  := A (J);
          A (J) := A (J + 1);
          A (J + 1) := Temp;
        end if;
      end loop;
    end loop;
  end Bubble_Sort;

  procedure Merge_Sort (A : in out Int_Array) is
    Temp : Int_Array (A'Range);

    procedure Merge (L, M, R : Integer) is
      I, J, K : Integer;
    begin
      I := L;
      J := M + 1;
      K := L;

      while I <= M and J <= R loop
        if A (I) < A (J) then
          Temp (K) := A (I);
          I  := I + 1;
        else
          Temp (K) := A (J);
          J  := J + 1;
        end if;
        K := K + 1;
      end loop;

      while I <= M loop
        Temp (K) := A (I);
        I  := I + 1;
        K  := K + 1;
      end loop;

      while J <= R loop
        Temp (K) := A (J);
        J   := J + 1;
        K  := K + 1;
      end loop;

      for I in L .. R loop
        A (I) := Temp (I);
      end loop;
    end Merge;
```

```
    procedure Merge_Sort_Recursive (L, R : Integer) is
      M : Integer;
    begin
      if L < R then
        M := (L + R) / 2;
        Merge_Sort_Recursive (L, M);
        Merge_Sort_Recursive (M + 1, R);
        Merge (L, M, R);
      end if;
    end Merge_Sort_Recursive;
  begin
    Merge_Sort_Recursive (A'First, A'Last);
  end Merge_Sort;
end t2_luka;

--t2_luka.ads
package t2_luka is
   type Int_Array is array (Integer range <>) of Integer;

   procedure Selection_Sort(A: in out Int_Array);
   procedure Insertion_Sort(A: in out Int_Array);
   procedure Bubble_Sort(A: in out Int_Array);
   procedure Merge_Sort(A: in out Int_Array);
end t2_luka;

--t2_lukamain.adb
with Ada.Text_IO;
with t2_luka;

use Ada.Text_IO;
use t2_luka;

procedure t2_lukamain is
   A : Int_Array(1 .. 6) := (3, 1, 7, 4, 9, 2);

   procedure Print_Array(A: Int_Array) is
   begin
      for I in A'First .. A'Last loop
         Put(Integer'Image(A(I)) & " ");
      end loop;
      New_Line;
   end Print_Array;
```

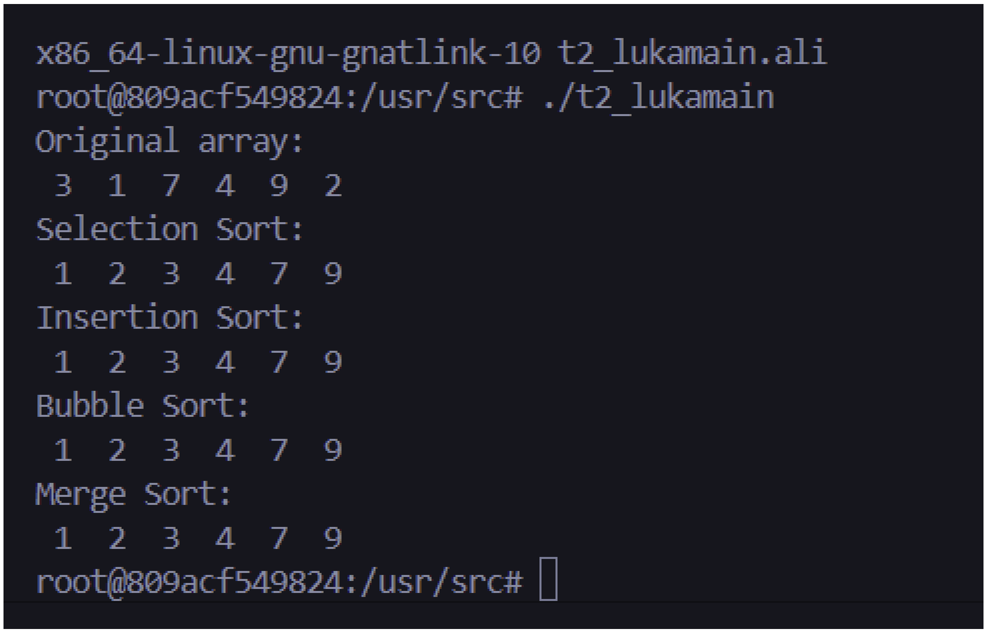
```
begin
  Put_Line("Original array:");
  Print_Array(A);

  Selection_Sort(A);
  Put_Line("Selection Sort:");
  Print_Array(A);

  A := (3, 1, 7, 4, 9, 2);
  Insertion_Sort(A);
  Put_Line("Insertion Sort:");
  Print_Array(A);

  A := (3, 1, 7, 4, 9, 2);
  Bubble_Sort(A);
  Put_Line("Bubble Sort:");
  Print_Array(A);

  A := (3, 1, 7, 4, 9, 2);
  Merge_Sort(A);
  Put_Line("Merge Sort:");
  Print_Array(A);
end t2_lukamain;
```



```
x86_64-linux-gnu-gnatlink-10 t2_lukamain.ali
root@809acf549824:/usr/src# ./t2_lukamain
Original array:
 3  1  7  4  9  2
Selection Sort:
 1  2  3  4  7  9
Insertion Sort:
 1  2  3  4  7  9
Bubble Sort:
 1  2  3  4  7  9
Merge Sort:
 1  2  3  4  7  9
root@809acf549824:/usr/src# 
```