Massachusetts Institute of Technology
Department of Aeronautics and Astronautics
Cambridge, MA 02139

Unified Engineering
Fall 2004

Problem Set #5
Solutions

PSET 5 Solution

CP11) Define an Ada95 record with four fields: **[5 points]**

a. Aircraft_ID of type integer

b. Airline of enumeration type with possible values {United, Delta, SouthWest, JetBlue, American}

c. Direction of type character

Aircraft_Type of type integer

```
type Airline_Type is
(United,  Delta_Airlines, Southwest,  Jetblue,  American);

 type Aircraft is  record
    Aircraft_Id   : Integer;
    Airline       : Airline_Type;
    Direction     : Character;
    Aircraft_Type : Integer;
end record;
```

CP12) **[35 points]**

Write an Ada95 program to:

i. Define four arrays with 5 elements each of types Aircraft_ID, Airline, Direction and Aircraft_Type

ii. Accept user input for 5 aircraft as shown in Figure 1 below.

iii. Sort the elements based on Aircraft_ID array as shown in Figure 2.

iv. Display the sorted arrays to the user

Aircraft_ID	**Airline**	**Direction**	**Aircraft_Type**
10	Delta	N	777
4	American	S	737
6	JetBlue	E	767
3	SouthWest	W	737
1	United	N	747

Figure 1: Four User Input Arrays

Aircraft_ID	**Airline**	**Direction**	**Aircraft_Type**
1	United	N	747

3	SouthWest	W	737
4	American	S	737
6	JetBlue	E	767
10	Delta	N	777

Figure 2: Sorted Arrays Displayed to the User

Turn in a **hard copy** of your **code listing** and an **electronic copy** of your **code**.

GNAT 3.14p (20010503) Copyright 1992-2001 Free Software Foundation, Inc.

Compiling:
c:/docume~1/kristina/mydocu~1/underv~1/unifie~2/fall20~1/pset/pset5~1/aircraft_using_arrays.adb
(source file time stamp: 2004-10-14 03:50:10)

```
 1. ---------------------------------------------------------------------
 2. -- Program to create four user input arrays of type aircraft_ID,
 3. -- Airline, Direction, and Aircraft_type. Sort the arrays based on
 4. -- the contents of the aircraft_id array and display the results to
 5. -- the user.
 6. -- Programmer : Joe B
 7. -- Date Created: October 10, 2004
 8. ---------------------------------------------------------------------
 9. with Ada.Text_Io;
10. with Ada.Integer_Text_Io;
11.
12. procedure Aircraft_Using_Arrays is
13.
14.    --aircraft ID array contains 5 integers
15.    type Aircraft_Id_Array is array (1 .. 5) of Integer;
16.
17.    -- airline is an enumeration type with 5 elements as shown below
18.    -- delta is a reserved word in Ada95 and hence we use Delta_Airlines
19.    type Airline is
20.         (United,
21.          Delta_Airlines,
22.          Southwest,
23.          Jetblue,
24.          American);
25.
26.    --define a new package called airline IO to carry out Airline specific IO
27.    package Airline_Io is new Ada.Text_Io.Enumeration_Io(Enum => Airline);
28.    --define an airline array of 5 elements
29.    type Airline_Array is array (1 .. 5) of Airline;
30.
31.    --define a direction array with 5 elements of type character
32.    type Direction_Array is array (1 .. 5) of Character;
33.
34.    --define a aircraft array of type integer
35.    type Aircraft_Type_Array is array (1 .. 5) of Integer;
36.
37.    --define four arrays to store the user input data
38.    My_Aircraft_Id        : Aircraft_Id_Array;
39.    My_Airline_Array      : Airline_Array;
```

```
   My_Direction_Array     : Direction_Array;
   My_Aircraft_Type_Array : Aircraft_Type_Array;

   --define four temporary variables for carrying out the swaps
   Temp_Aircraft_Id   : Integer;
   Temp_Aircraft_Type : Integer;
   Temp_Direction     : Character;
   Temp_Airline       : Airline;

begin
   --get 5 inputs from the user
   for I in 1 .. 5 loop
      Ada.Text_Io.Put("For Aircraft ");
      Ada.Text_Io.Put_Line(Integer'Image(I));

      Ada.Text_Io.Put("Please enter Aircraft ID : ");
      Ada.Integer_Text_Io.Get(My_Aircraft_Id(I));
      Ada.Text_Io.Skip_Line;

      Ada.Text_Io.Put("Please enter Airline: ");
      Airline_Io.Get(My_Airline_Array(I));
      Ada.Text_Io.Skip_Line;

      Ada.Text_Io.Put("Please enter Aircraft Direction: ");
      Ada.Text_Io.Get(My_Direction_Array(I));
      Ada.Text_Io.Skip_Line;

      Ada.Text_Io.Put("Please enter Aircraft Type: ");
      Ada.Integer_Text_Io.Get(My_Aircraft_Type_Array(I));
      Ada.Text_Io.Skip_Line;
   end loop;

   -- display input elements to the user
   Ada.Text_Io.Put_Line("Unsorted Array is: ");
   for I in 1 .. 5 loop
      Ada.Text_Io.Put_Line("Aircraft ID     Airline     Direction     Type");
      Ada.Text_Io.Put(Integer'Image(My_Aircraft_Id(I)));
      Ada.Text_Io.Put("      ");
      Ada.Text_Io.Put(Airline'Image(My_Airline_Array(I)));
      Ada.Text_Io.Put("      ");
      Ada.Text_Io.Put(My_Direction_Array(I));
      Ada.Text_Io.Put("      ");
      Ada.Text_Io.Put(Integer'Image(My_Aircraft_Type_Array(I)));
      Ada.Text_Io.Put("      ");
      Ada.Text_Io.New_Line;
   end loop;
   Ada.Text_Io.New_Line;

   --sort the elements based on aircraft_id
   for I in 1 .. 4 loop
      for J in I+1 .. 5 loop
         if My_Aircraft_Id(I)> My_Aircraft_Id(J) then
            --store the temporary values
            Temp_Aircraft_Id := My_Aircraft_Id(I);
            Temp_Aircraft_Type := My_Aircraft_Type_Array(I);
            Temp_Airline := My_Airline_Array(I);
            Temp_Direction := My_Direction_Array(I);

```

```
98.          -- store the values in jth location into ith location
99.          My_Aircraft_Id(I):= My_Aircraft_Id(J);
100.          My_Aircraft_Type_Array(I):= My_Aircraft_Type_Array(J);
101.          My_Airline_Array(I):= My_Airline_Array(J);
102.          My_Direction_Array(I):= My_Direction_Array(J);
103.
104.          --store the values in temp into the jth loaction
105.          My_Aircraft_Id(J):=Temp_Aircraft_Id ;
106.           My_Aircraft_Type_Array(J):=Temp_Aircraft_Type;
107.          My_Airline_Array(J) := Temp_Airline;
108.          My_Direction_Array(J) := Temp_Direction;
109.        end if;
110.      end loop;
111.    end loop;
112.
113.    -- display the sorted arrays to the user
114.    Ada.Text_Io.Put_Line("Sorted Array is: ");
115.    for I in 1 .. 5 loop
116.      Ada.Text_Io.Put_Line("Aircraft ID     Airline     Direction     Type");
117.      Ada.Text_Io.Put(Integer'Image(My_Aircraft_Id(I)));
118.      Ada.Text_Io.Put("      ");
119.      Ada.Text_Io.Put(Airline'Image(My_Airline_Array(I)));
120.      Ada.Text_Io.Put("      ");
121.      Ada.Text_Io.Put(My_Direction_Array(I));
122.      Ada.Text_Io.Put("      ");
123.      Ada.Text_Io.Put(Integer'Image(My_Aircraft_Type_Array(I)));
124.      Ada.Text_Io.Put("      ");
125.      Ada.Text_Io.New_Line;
126.    end loop;
127.
128. end Aircraft_Using_Arrays;
129.
```

CP13) **[60 points]**

a. Define your own package, with the type definition of an array with ten aircraft records (as defined in question 1 above), and four functions/procedures to:

   i. Sort the array in ascending order based on Aircraft_ID.

   ii. Display the contents of records in the array

   iii. Read in aircraft information into the array based on user input.

   iv. Read in aircraft information into the array from an input file called aircraft_record_input.txt

b. Write a test program that uses your package to:

   i. Create an array with 5 records input by the user, and 5 records read in from the file called aircraft_record_input.txt

ii. Sort the records based on the Aircraft_Id

iii. Display each record of the array as shown in Figure 3.

```
Aircraft_ID:   1
Airline:       United
Direction:     N
Aircraft_Type: 747
…
…
Aircraft_ID:   10
Airline:       Delta
Direction:     N
Aircraft_Type: 777
```

Figure 3. Sorted Records Displayed to User

iv. Store the contents into a sorted array called sorted_aircraft_records.txt

Turn in a **hard copy** of your **code listing** and an **electronic copy** of your **code**.

GNAT 3.14p (20010503) Copyright 1992-2001 Free Software Foundation, Inc.

Checking:
c:/docume~1/kristina/mydocu~1/underv~1/unifie~2/fall20~1/pset/pset5~1/my_aircraft_package.ads
(source file time stamp: 2004-10-14 03:49:52)

```
 1. package My_Aircraft_Package is
 2.
 3.    -- airline_type is an enumeration type with 5 elements as shown below
 4.    -- delta is a reserved word in Ada95 and hence we use Delta_Airlines
 5.    type Airline_Type is
 6.          (United,
 7.           Delta_Airlines,
 8.           Southwest,
 9.           Jetblue,
10.        American);
11.
12.    -- aircraft record is created with four fields as shown below
13.    type Aircraft is
14.       record
15.          Aircraft_Id   : Integer;
16.          Airline       : Airline_Type;
17.          Direction     : Character;
18.          Aircraft_Type : Integer;
19.       end record;
20.
21.    --define an aircraft array of 10 elements
22.    type Aircraft_Array is array (1 .. 10) of Aircraft;
23.
24.    procedure Get_Aircraft_From_Array (
25.          My_Aircraft_Array : in out Aircraft_Array );
26.
27.    procedure Get_Aircraft_From_File (
28.          My_Aircraft_Array : in out Aircraft_Array );
29.
30.    procedure Display_Array (
31.          My_Aircraft_Array : in     Aircraft_Array );
32.
33.    procedure Sort_Array (
34.       My_Aircraft_Array : in out Aircraft_Array ) ;
35.
36.    procedure Store_Aircraft_In_File (
37.       My_Aircraft_Array : in     Aircraft_Array );
38.
39. end My_Aircraft_Package;
40.
```

40 lines: No errors

GNAT 3.14p (20010503) Copyright 1992-2001 Free Software Foundation, Inc.

Compiling:
c:/docume~1/kristina/mydocu~1/underv~1/unifie~2/fall20~1/pset/pset5~1/my_aircraft_package.adb
(source file time stamp: 2004-10-14 03:50:42)

```
 1. with Ada.Text_Io;
 2. with Ada.Integer_Text_Io;
 3.
 4. package body My_Aircraft_Package is
 5.    --define a new package called airline IO to carry out Airline specific IO
 6.    package Airline_Io is new Ada.Text_Io.Enumeration_Io(Enum => Airline_Type);
 7.
 8.    --procedure to get user input into array
 9.    procedure Get_Aircraft_From_Array (
10.          My_Aircraft_Array : in out Aircraft_Array ) is
11.    begin
12.       --get 5 inputs from the user
13.       for I in 1 .. 5 loop
14.          Ada.Text_Io.Put("For Aircraft ");
15.          Ada.Text_Io.Put_Line(Integer'Image(I));
16.
17.          Ada.Text_Io.Put("Please enter Aircraft ID : ");
18.          Ada.Integer_Text_Io.Get(My_Aircraft_Array(I).Aircraft_Id);
19.          Ada.Text_Io.Skip_Line;
20.
21.          Ada.Text_Io.Put("Please enter Airline: ");
22.          Airline_Io.Get(My_Aircraft_Array(I).Airline);
23.          Ada.Text_Io.Skip_Line;
24.
25.          Ada.Text_Io.Put("Please enter Aircraft Direction: ");
26.          Ada.Text_Io.Get(My_Aircraft_Array(I).Direction);
27.          Ada.Text_Io.Skip_Line;
28.
29.          Ada.Text_Io.Put("Please enter Aircraft Type: ");
30.          Ada.Integer_Text_Io.Get(My_Aircraft_Array(I).Aircraft_Type);
31.          Ada.Text_Io.Skip_Line;
32.       end loop;
33.    end Get_Aircraft_From_Array;
34.
35.    -- procedure to get 5 records from a file.
36.    -- notes:
37.    --1. you will have to have an extra line after the inputs if you
38.    --   use the skip_line. An alternative is to check for end_of_File
39.    --   before the last skip_line in the loop.
40.    --2. The input file has to be in the same folder as the code
41.    --3. The output file is created in the same folder as the code
42.    procedure Get_Aircraft_From_File (
43.          My_Aircraft_Array : in out Aircraft_Array ) is
44.       My_File : Ada.Text_Io.File_Type;
45.    begin
46.       Ada.Text_Io.Open(My_File, Ada.Text_Io.In_File, "aircraft_record_input.txt");
47.
48.       --get 5 inputs from the file
49.       for I in 6 .. 10 loop
50.          Ada.Integer_Text_Io.Get(My_File, My_Aircraft_Array(I).Aircraft_Id);
51.          Ada.Text_Io.Skip_Line(My_File);
```

```
52.         Airline_Io.Get(My_File, My_Aircraft_Array(I).Airline);
53.         Ada.Text_Io.Skip_Line(My_File);
54.         Ada.Text_Io.Get(My_File, My_Aircraft_Array(I).Direction);
55.         Ada.Text_Io.Skip_Line(My_File);
56.         Ada.Integer_Text_Io.Get(My_File, My_Aircraft_Array(I).Aircraft_Type);
57.         Ada.Text_Io.Skip_Line(My_File);
58.      end loop;
59.      Ada.Text_Io.Close(My_File);
60.
61.   end Get_Aircraft_From_File;
62.
63.   -- procedure to display contents of the array to the user
64.   procedure Display_Array (
65.         My_Aircraft_Array : in     Aircraft_Array ) is
66.   begin
67.      -- display the array to the user
68.      for I in 1 .. 10 loop
69.         Ada.Text_Io.Put_Line("ID      Airline      Direction      Type");
70.         Ada.Text_Io.Put(Integer'Image(My_Aircraft_Array(I).Aircraft_Id));
71.         Ada.Text_Io.Put("      ");
72.         Ada.Text_Io.Put(Airline_Type'Image(My_Aircraft_Array(I).Airline));
73.         Ada.Text_Io.Put("      ");
74.         Ada.Text_Io.Put(My_Aircraft_Array(I).Direction);
75.         Ada.Text_Io.Put("      ");
76.         Ada.Text_Io.Put(Integer'Image(My_Aircraft_Array(I).Aircraft_Type));
77.         Ada.Text_Io.Put("      ");
78.         Ada.Text_Io.New_Line;
79.      end loop;
80.   end Display_Array;
81.
82.   --procedure to sort the array
83.   procedure Sort_Array (
84.         My_Aircraft_Array : in out Aircraft_Array ) is
85.      Temp : Aircraft;
86.   begin
87.      --sort the elements based on aircraft_id
88.      for I in 1 .. 9 loop
89.         for J in I+1 .. 10 loop
90.            if My_Aircraft_Array(I).Aircraft_Id> My_Aircraft_Array(J).Aircraft_Id then
91.               Temp:=My_Aircraft_Array(I);
92.               My_Aircraft_Array(I) := My_Aircraft_Array(J);
93.               My_Aircraft_Array(J):= Temp;
94.            end if;
95.         end loop;
96.      end loop;
97.   end Sort_Array;
98.
99.   --procedure to store the sorted array into the file
100.   procedure Store_Aircraft_In_File (
101.         My_Aircraft_Array : in     Aircraft_Array ) is
102.      My_File : Ada.Text_Io.File_Type;
103.   begin
104.      Ada.Text_Io.Create(My_File, Ada.Text_Io.Out_File, "aircraft_record_output.txt");
105.      --Ada.Text_Io.Open(My_File, Ada.Text_Io.Out_File, "aircraft_record_output.txt");
106.
107.      --get 5 inputs from the file
108.      for I in 1 .. 10 loop
109.         Ada.Integer_Text_Io.Put(My_File, My_Aircraft_Array(I).Aircraft_Id);
```

```
110.          Ada.Text_Io.New_Line(My_File);
111.          Airline_Io.Put(My_File, My_Aircraft_Array(I).Airline);
112.          Ada.Text_Io.New_Line(My_File);
113.          Ada.Text_Io.Put(My_File, My_Aircraft_Array(I).Direction);
114.          Ada.Text_Io.New_Line(My_File);
115.          Ada.Integer_Text_Io.Put(My_File, My_Aircraft_Array(I).Aircraft_Type);
116.          Ada.Text_Io.New_Line(My_File);
117.       end loop;
118.       Ada.Text_Io.Close(My_File);
119.
120.    end Store_Aircraft_In_File;
121. end My_Aircraft_Package;
122.
```

122 lines: No errors

GNAT 3.14p (20010503) Copyright 1992-2001 Free Software Foundation, Inc.

Compiling:
c:/docume~1/kristina/mydocu~1/underv~1/unifie~2/fall20~1/pset/pset5~1/aircraft_using_records.adb
(source file time stamp: 2004-10-14 03:50:26)

```
 1. ------------------------------------------------------------------
 2. -- Program to create an array of 10 records of type aircraft, half
 3. -- of which are entered by the user and the remaining are read in
 4. -- from a file called aircraft_input_records.txt
 5. -- Programmer : Joe B
 6. -- Date Created: October 10, 2004
 7. ------------------------------------------------------------------
 8. with Ada.Text_Io;
 9. with My_Aircraft_Package;
10. use My_Aircraft_Package;
11.
12. procedure Aircraft_Using_Records is
13.
14.    My_Aircraft_Array : My_Aircraft_Package.Aircraft_Array;
15.
16. begin
17.    Get_Aircraft_From_Array(My_Aircraft_Array);
18.    Get_Aircraft_From_File(My_Aircraft_Array);
19.    Ada.Text_Io.Put_Line("Unsorted Array");
20.    Ada.Text_Io.New_Line;
21.    Display_Array(My_Aircraft_Array);
22.    Sort_Array(My_Aircraft_Array);
23.    Ada.Text_Io.New_Line;
24.    Ada.Text_Io.Put_Line("Sorted Array");
25.    Ada.Text_Io.New_Line;
26.    Display_Array(My_Aircraft_Array);
27.    Store_Aircraft_In_File(My_Aircraft_Array);
28. end Aircraft_Using_Records;
29.
```

29 lines: No errors

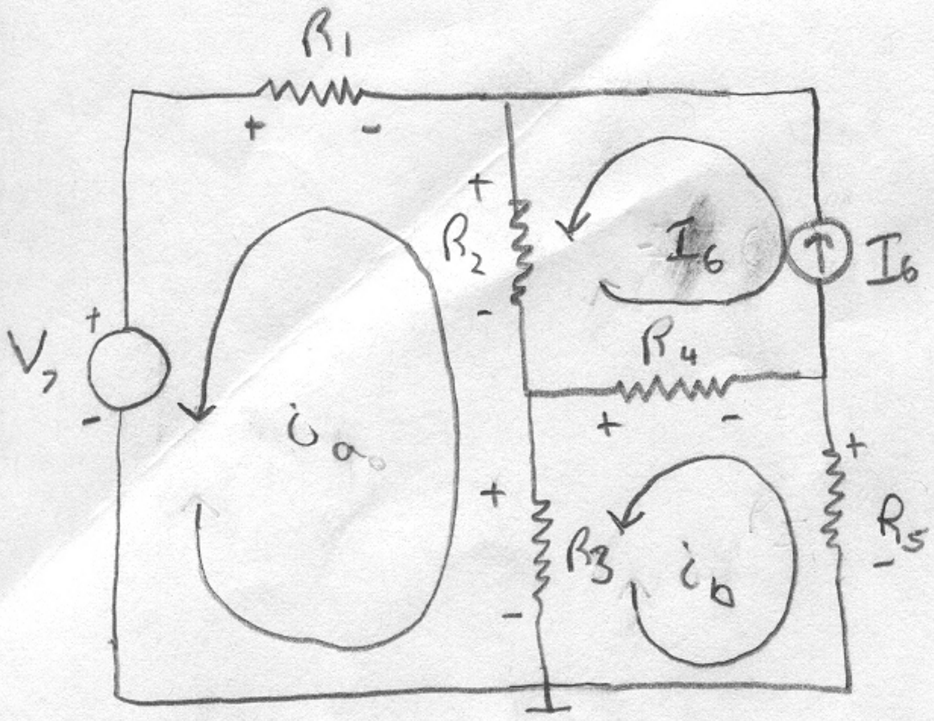
aircraft_record_input.txt

```
120
SouthWest
N
737
110
JetBlue
S
737
130
American
W
777
121
United
E
767
140
JetBlue
S
767
```

aircraft_record_output.txt

```
 1
UNITED
N
 747
 3
SOUTHWEST
W
 737
 4
AMERICAN
S
 737
 6
JETBLUE
E
 767
 10
DELTA_AIRLINES
N
 777
 110
JETBLUE
S
 737
 120
SOUTHWEST
N
 737
 121
UNITED
E
 767
 130
AMERICAN
W
 777
 140
JETBLUE
S
 767
```

Unified Engineering
S4 Solutions



=

values from problem

$R_1 = 1\,\Omega$

$R_2 = 2\,\Omega$

$R_3 = 4\,\Omega$

$R_4 = 6\,\Omega$

$R_5 = 1\,\Omega$

$I_6 = 3\,A$

$V_7 = 5\,V$

setting up loop eens.

for i_a:

$$-V_7 + i_a R_1 + (i_a - I_6) R_2 + (i_a - i_b) R_3 = 0$$

for i_b:

$$(i_b - i_a) R_3 + (i_b - I_6) R_4 + i_b R_5 = 0$$

gathering terms

$$(R_1 + R_2 + R_3)\, i_a - R_3 i_b = I_6 R_2 - V_7$$

$$-R_3 i_a + (R_3 + R_4 + R_5)\, i_b = I_6 R_4$$

pluging in given terms

$$7 i_a - 4 i_b = 1 \quad (1)$$

$$-4 i_a + 11 i_b = 18 \quad (2)$$

Solving these eans.

multiply (1) by 4 or (2) by 4

$$28 i_a - 16 i_b = 4$$

$$-28 i_a + 77 i_b = 126$$

$$61 i_b = 130$$

so $\underline{i_b = 130/61}$

$$7 i_a - 4(130/61) = 1$$

$$7 i_a + 8 = 1$$

so $\underline{i_a = -83/61}$

so the voltages for all the componets are:

$V_1 = -i_a R_1 =$ $\boxed{-83/61\ V}$ -1.36

$V_2 = -(i_a - I_6) R_2 =$ $\boxed{200/61\ V}$ 3.28

$V_3 = -(i_a - i_5) R_3 =$ $\boxed{187/61\ V}$ 3.08

$V_4 = -(i_5 - I_6) R_4 =$ $\boxed{318/61\ V}$ 5.22

$V_5 = -i_5 R_5 =$ $\boxed{-130/61\ V}$ -2.13

$V_6 = e_1 - e_3$ but we don't know e_1 or e_3 so we find them using known currents $-[(i_a R_1 + V_7) + i_5 R_5] =$

$\boxed{-518/61\ V}$ -8.49

$V_7 = \boxed{5\ V}$

The currents are

$i_1 = -i_a =$ $\boxed{-83/61\ A}$ -1.36

$i_2 = -(i_a - I_6) =$ $\boxed{100/61\ A}$ 1.64

$i_3 = -(i_a - i_5) =$ $\boxed{47/61\ A}$ $.77$

$i_4 = -(i_a - I_6) =$ $\boxed{53/61\ A}$ $.87$

$i_5 = -i_a =$ $\boxed{-130/61\ A}$ -2.13

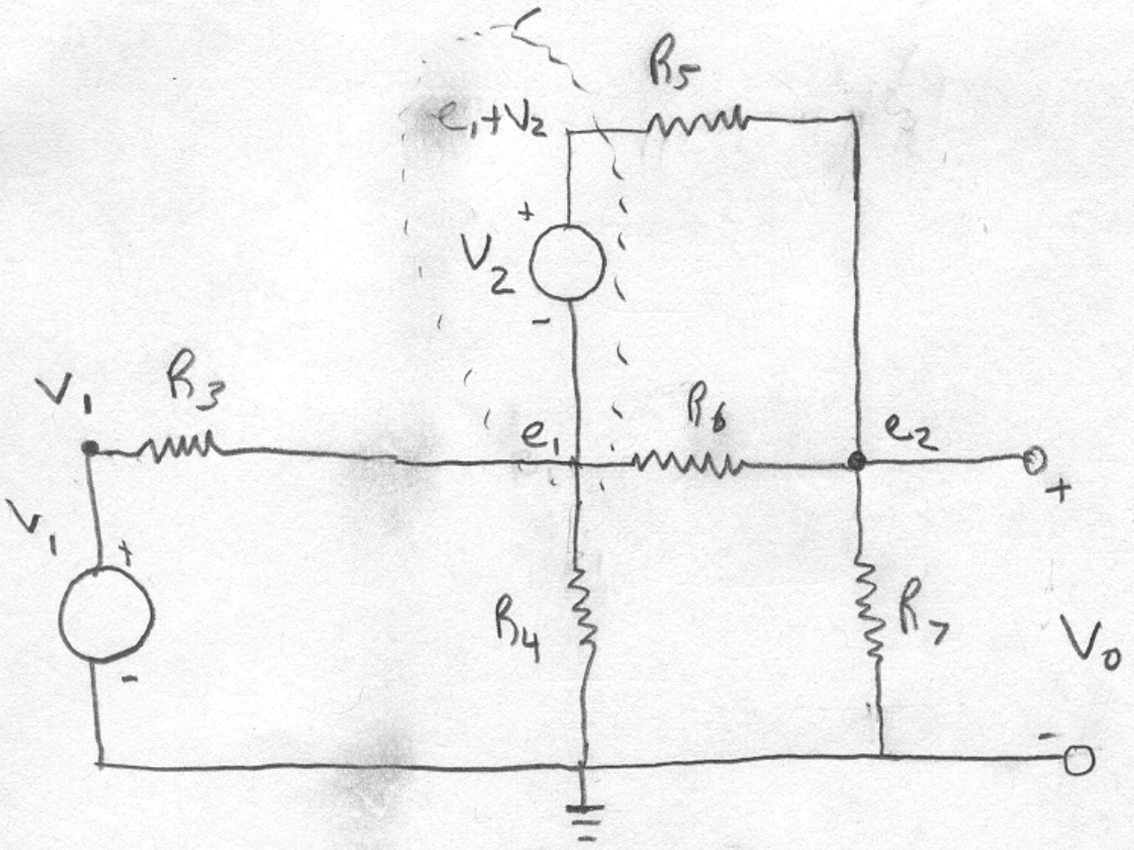
$I_6 = 3A$

$i_7 = i_a =$ $\boxed{83/61\ A}$ 1.36

Unified Engineering

Fall 200

5.5 Solutions



Given values:

$R_3 = 2\ \Omega$

$R_4 = 2\ \Omega$

$R_5 = 8\ \Omega$

$R_6 = 8\ \Omega$

$R_7 = 3\ \Omega$

$V_1 = 10\ V$

$V_2 = 5\ V$

use node method

e_1 supernode:

$$\frac{(e_1+V_2-e_2)}{R_5} + \frac{e_1-e_2}{R_6} + \frac{e_1-V_1}{R_3} + \frac{e_1}{R_4} = 0$$

node e_2:

$$\frac{e_2-(e_1+V_2)}{R_5} + \frac{e_2-e_1}{R_6} + \frac{e_2}{R_7}$$

gathering terms

$$(G_3+G_4+G_5+G_6)e_1 - G_6 e_2 = -G_5 V_2 + G_3 V_1$$

$$-(G_5+G_6)e_1 + (G_5+G_6+G_7)e_2 = G_5 V_2$$

plug in numbers

$$5/4\, e_1 - 1/8\, e_2 = 35/8 \quad (1)$$

$$-1/4\, e_1 + 7/12\, e_2 = 5/8 \quad (2)$$

solve for e_2

multiply (2) by 5

$$5/4\, e_1 - 1/8\, e_2 = 35/8$$

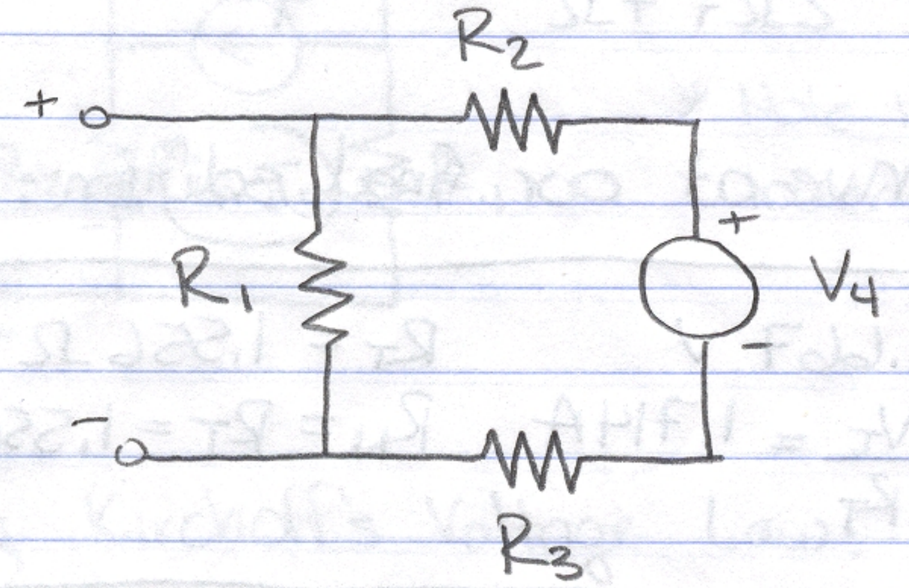
$$-5/4\, e_1 + 35/12\, e_2 = 25/8$$

$$67/24\, e_2 = 15/2 \quad \Rightarrow \text{so} \quad \boxed{e_2 = 180/64 = V_o}$$

Problem Set #(5) SOLUTION

Problem 5.6

1.

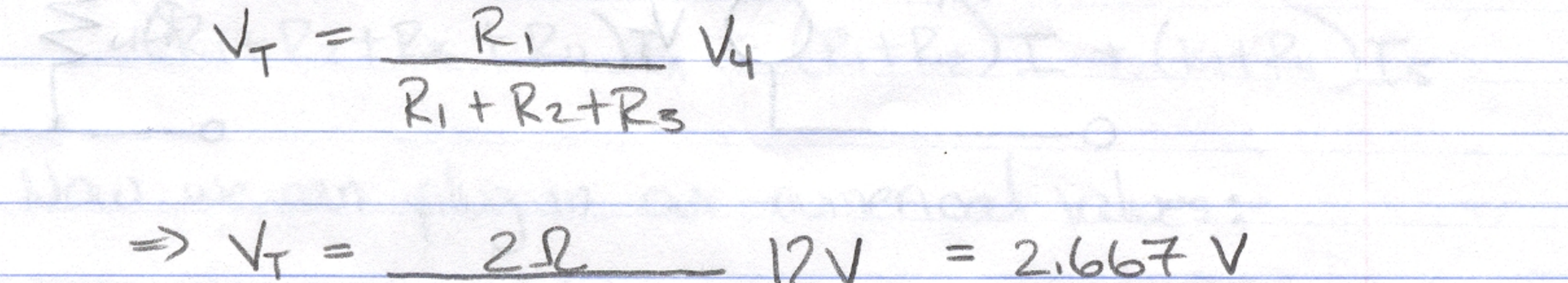


Because this circuit is a voltage divider, we can calculate the open-circuit voltage directly. We have that:

$$V_T = \frac{R_1}{R_1+R_2+R_3} V_4$$

$$\Rightarrow V_T = \frac{2\Omega}{2\Omega+4\Omega+3\Omega} 12V = \underline{\underline{2.667\ V}}$$

We can find R_T by setting all voltage sources to zero:



$$\Rightarrow R_T = R_1 \| (R_2 + R_3) = 2\Omega \| (4\Omega + 3\Omega) = 2\Omega \| 7\Omega$$

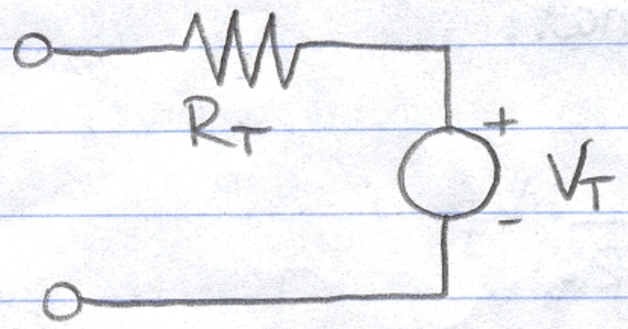
$$= \frac{(2\Omega)(7\Omega)}{2\Omega + 7\Omega} = \frac{14}{9}\Omega = \underline{1.556\ \Omega}$$

So we arrive at our final solution:

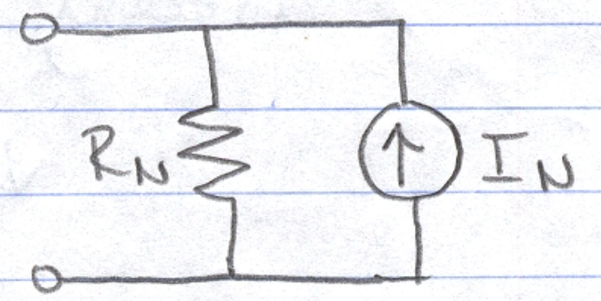
$V_T = 2.667\ V$ $\quad R_T = 1.556\ \Omega$

$I_N = \frac{V_T}{R_T} = 1.714 A$ $\quad R_N = R_T = 1.556\ \Omega$

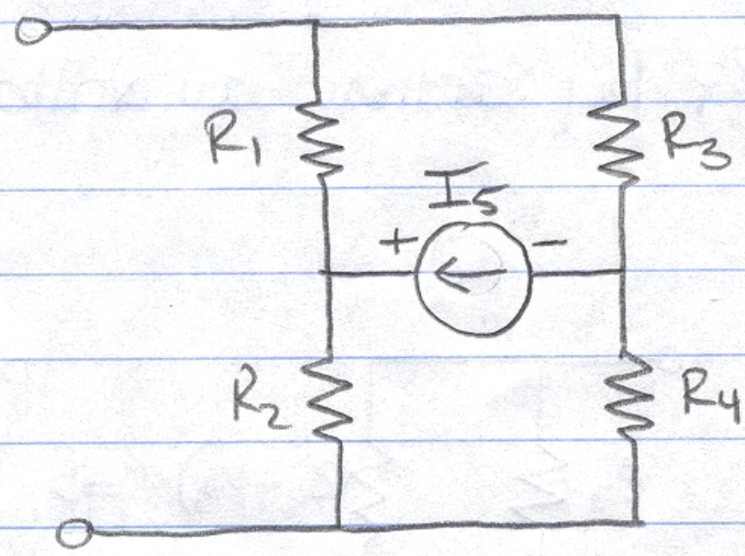
Thevinin Equivalent Circuit



Norton Equivalent Circuit

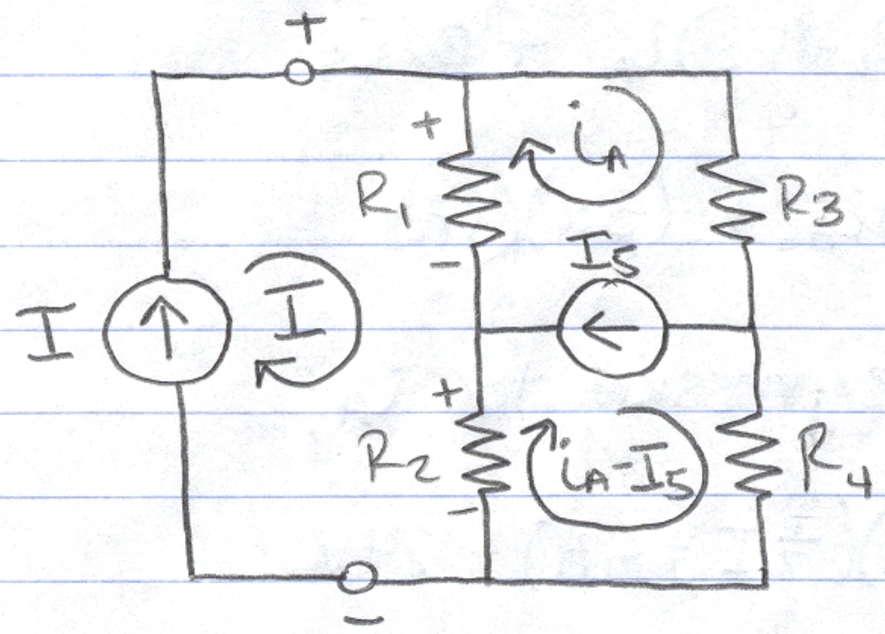


2.



The most convenient method of solving this circuit is to add a test current source.

We can then solve directly using the loop method.



* Note the labeling of the lower-right loop current!

Applying Kirchoff's Voltage Law:

$$i_A: \quad R_1(i_A - I) + R_3 i_A - V_5 = 0$$
$$i_A - I_5: \quad R_2(i_A - I_5 - I) + V_5 + R_4(i_A - I_5) = 0$$

Eliminating V_5,

$$(R_1 + R_2 + R_3 + R_4) i_A = (R_1 + R_2) I + (R_2 + R_4) I_5$$

Now we can plug in our numerical values:

$$(1\Omega + 3\Omega + 3\Omega + 1\Omega) i_A = (1\Omega + 3\Omega) I + (3\Omega + 1\Omega)(8A)$$

$$\Rightarrow 8 i_A = 4I + 32A$$

$$\Rightarrow i_A = \frac{1}{2} I + 4A$$

We can now solve for the voltage across the terminals:

$$V = V_1 + V_2 = R_1(I - i_A) + R_2(I - (i_A - I_5))$$

$$= (R_1+R_2)I - (R_1+R_2)i_A + R_2 I_S$$

$$= (4\Omega)I - (4\Omega)i_A + (3\Omega)(8A)$$

Plugging in our expression for i_A,

$$V = (4\Omega)I - (4\Omega)\left(\tfrac{1}{2}I + 4A\right) + 24V$$

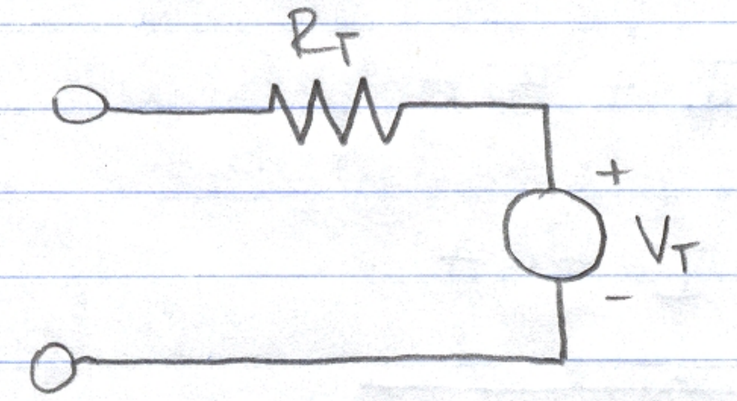
$$= \underbrace{(2\Omega)}_{R_T} I + \underbrace{8V}_{V_T}$$

So we arrive at our final solution:

$$R_T = R_N = 2\Omega \qquad V_T = 8V$$

$$I_N = \frac{V_T}{R_T} = 4A$$

Thevinin Equivalent Circuit



Norton Equivalent Circuit