Programming Fundamental

Week 10 –
Selective control Structures in the C Programming Language

Dr. Maria Irmina Prasetiyowati, S.Kom,. M.T.
Alethea Suryadibrata, S.Kom., M.Eng.
Putri Sanggabuana Setiawan, S.Kom, M.T.I.
Januar Wahjudi, S.Kom., M.Sc.
Drs Slamet Aji Pamungkas, M.Eng
Kursehi Falgenti S.Kom., M.Kom.

Weekly Learning Outcomes for Subjects (Sub-CPMK):

Sub-CPMK 0614: Students are able to create simple programs with elements of selection control, repetition control, functions or procedures, and implement arrays and pointers in the C programming language (C6).

Outline

1. IF…
2. Nested IF
3. Switch Case

Control Structures

- Control the flow of execution in a program or function
- Combine individual instructions into a single logical unit with one entry point and one exit point
- Instructions are organized into 3 kinds of control structures to control execution flow
  - Sequence
  - Selection
  - Repetition

Sequential Flow

- A compound statement is used to specify sequential flow
- A compound statement is written as a group of statements bracketed by { and }
- Control flows from $statement_1$ to $statement_2$, and so on

```
{
        statement1;
        statement2;
            .
            .
            .
        statementn;
}
```

Selection Control Structures

- A selection control structure chooses which alternative to execute

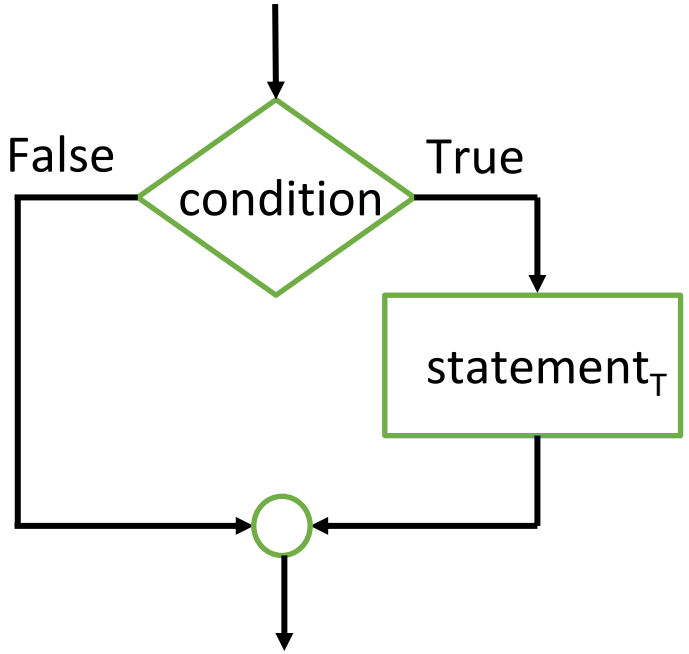
  - if statement
  - switch statement

- A program chooses among alternative statements by testing the value of key variables
- A **condition** establishes a criterion for either executing or skipping a group of statements
- A **condition** is an expression that is either false (represented by 0) or true (usually represented by 1)

If Statement with One Alternative

```
if(condition)
        statement_T;
```

- Executes **only when the condition is true**




```c
#include <stdio.h>

int main()
{
    int grade;

    printf("Grade: ");
    scanf("%d",&grade);
    if(grade >= 55)
        printf("Passed\n");

    return 0;
}
```

F:\if.exe

```
Grade: 55
Passed

Process returned 0 (0x0)
execution time : 1.532 s
Press any key to continue.
```

F:\if.exe

```
Grade: 45

Process returned 0 (0x0)
execution time : 1.108 s
Press any key to continue.
```

If Statement with Two Alternatives

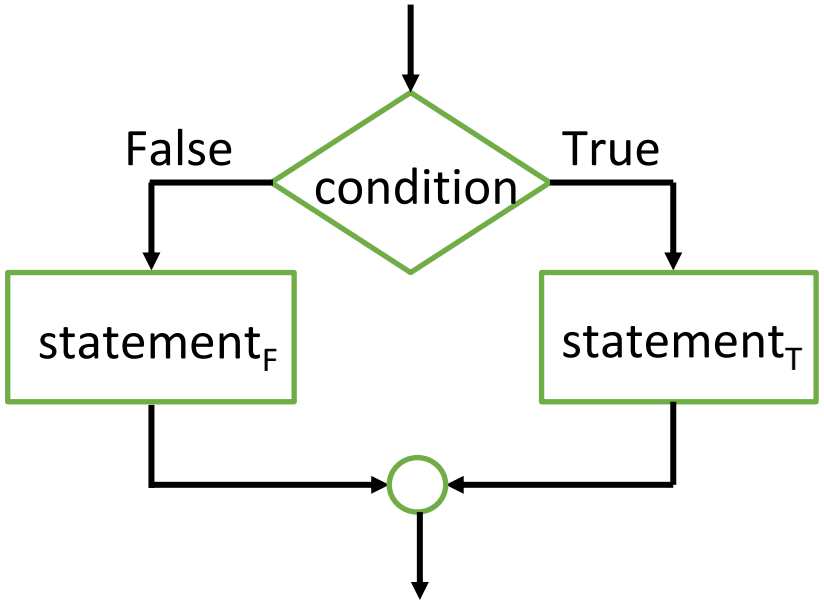
```
if(condition)
        statement_T;
else
        statement_F;
```

- Selects the statement following the parenthesized condition ($statement_T$) if the condition evaluates to 1 (true)
- Selects the statement following *else* ($statement_F$) if the condition evaluates to 0 (false)

```
if(grade >= 55)
    printf("Passed\n");
else
    printf("Failed\n");
```

```
F:\if.exe
Grade: 55
Passed

Process returned 0 (0x0)
execution time : 1.079 s
Press any key to continue.
```

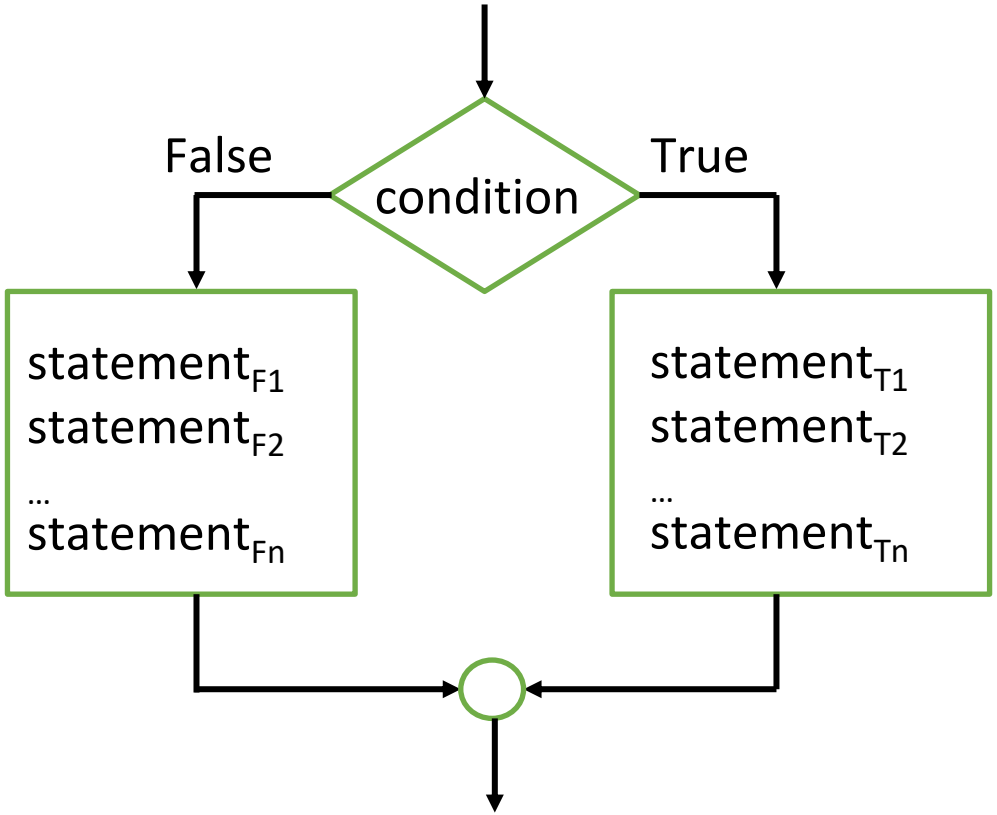
```
F:\if.exe
Grade: 45
Failed

Process returned 0 (0x0)
execution time : 1.322 s
Press any key to continue.
```



If Statement with Compound Statements

```
if(condition)
{
        statement_T1;
        statement_T2;
        ...
        statement_Tn;
}
else
{
        statement_F1;
        statement_F2;
        ...
        statement_Fn;
}
```

- When the symbol **{** follows the condition or *else*, the C compiler either executes or skips **all statements** through the matching **}**



If Statement with Compound Statements

```
if(condition)
{
        statement_T1;
        statement_T2;
        ...
        statement_Tn;
}
else
{
        statement_F1;
        statement_F2;
        ...
        statement_Fn;
}
```

```
if(grade >= 55)
    printf("Passed\n");
else
{
    printf("Failed\n");
    printf("You must take this course again\n");
}
```

F:\if.exe

```
Grade: 55
Passed

Process returned 0 (0x0)
execution time : 1.250 s
Press any key to continue.
```

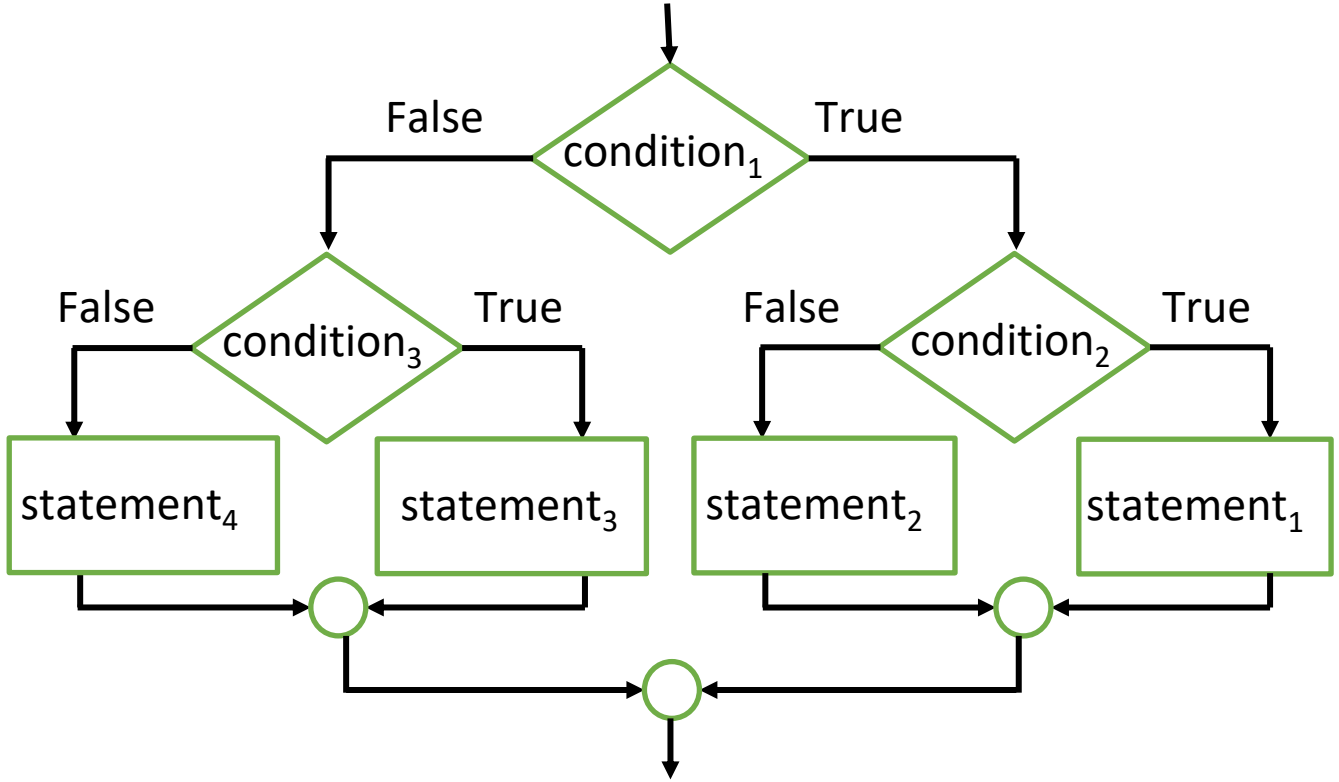
F:\if.exe

```
Grade: 45
Failed
You must take this course again

Process returned 0 (0x0)   execution time : 1.330 s
Press any key to continue.
```

Nested If Statement

```
if(condition1)
{
        if(condition2)
                statement1;
        else
                statement2;
}
else
{
        if(condition3)
                statement3;
        else
                statement4;
}
```

- An if statement with another if statement as its true task or its false task
- One if statement inside another



Nested If Statement

```
if(condition1)
{
        if(condition2)
                statement1;
        else
                statement2;
}
else
{
        if(condition3)
                statement3;
        else
                statement4;
}
```

```
if(systolicBloodPressure > 140)
    printf("Hypertension\n");
else
{
    if(systolicBloodPressure > 120)
        printf("Pre-hypertension\n");
    else
    {
        if(systolicBloodPressure > 90)
            printf("Normal\n");
        else
            printf("Hypotension\n");
    }
}
```

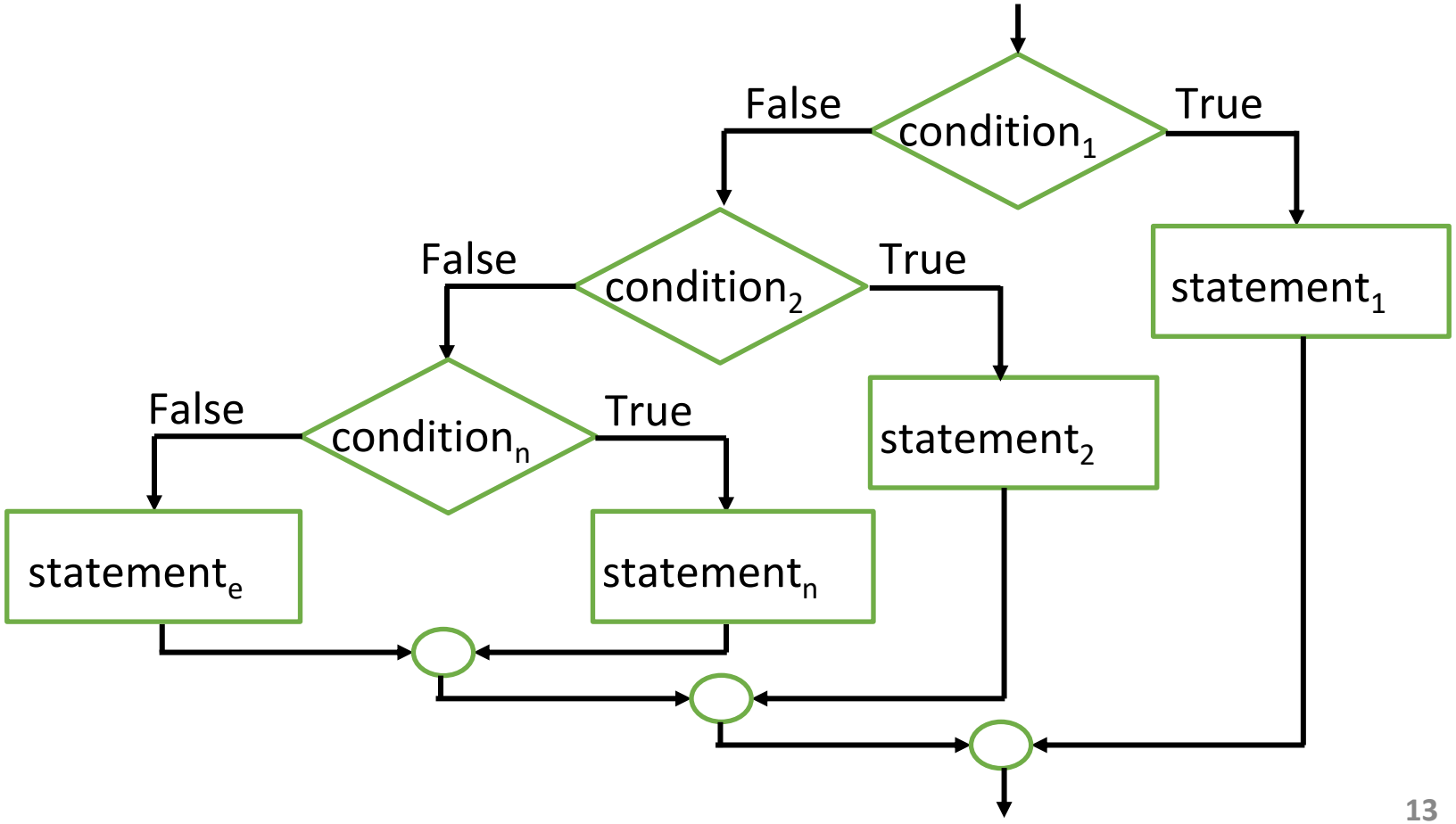
F:\if.exe

```
Systolic Blood Pressure: 80
Hypotension

Process returned 0 (0x0)   execution time : 1.395 s
Press any key to continue.
```

Multiple-Alternative Decisions

Nested if statements can become quite complex.
If there are more than 3 alternatives and indentation is not consistent, it may be difficult to determine the logical structure of the if statement

```
if(condition1)
        statement1;
else if(condition2)
        statement2;
              .
              .
              .
else if(conditionn)
        statementn;
else
        statemente;
```



Multiple-Alternative Decisions

```
if(condition1)
      statement1;
else if(condition2)
      statement2;
            .
            .
            .
else if(conditionn)
      statementn;
else
      statemente;
```

```
if(systolicBloodPressure > 140)
    printf("Hypertension\n");
else if(systolicBloodPressure > 120)
    printf("Pre-hypertension\n");
else if(systolicBloodPressure > 90)
    printf("Normal\n");
else
    printf("Hypotension\n");
```

```
F:\if.exe
Systolic Blood Pressure: 110
Normal

Process returned 0 (0x0)   execution time : 1.290 s
Press any key to continue.
```

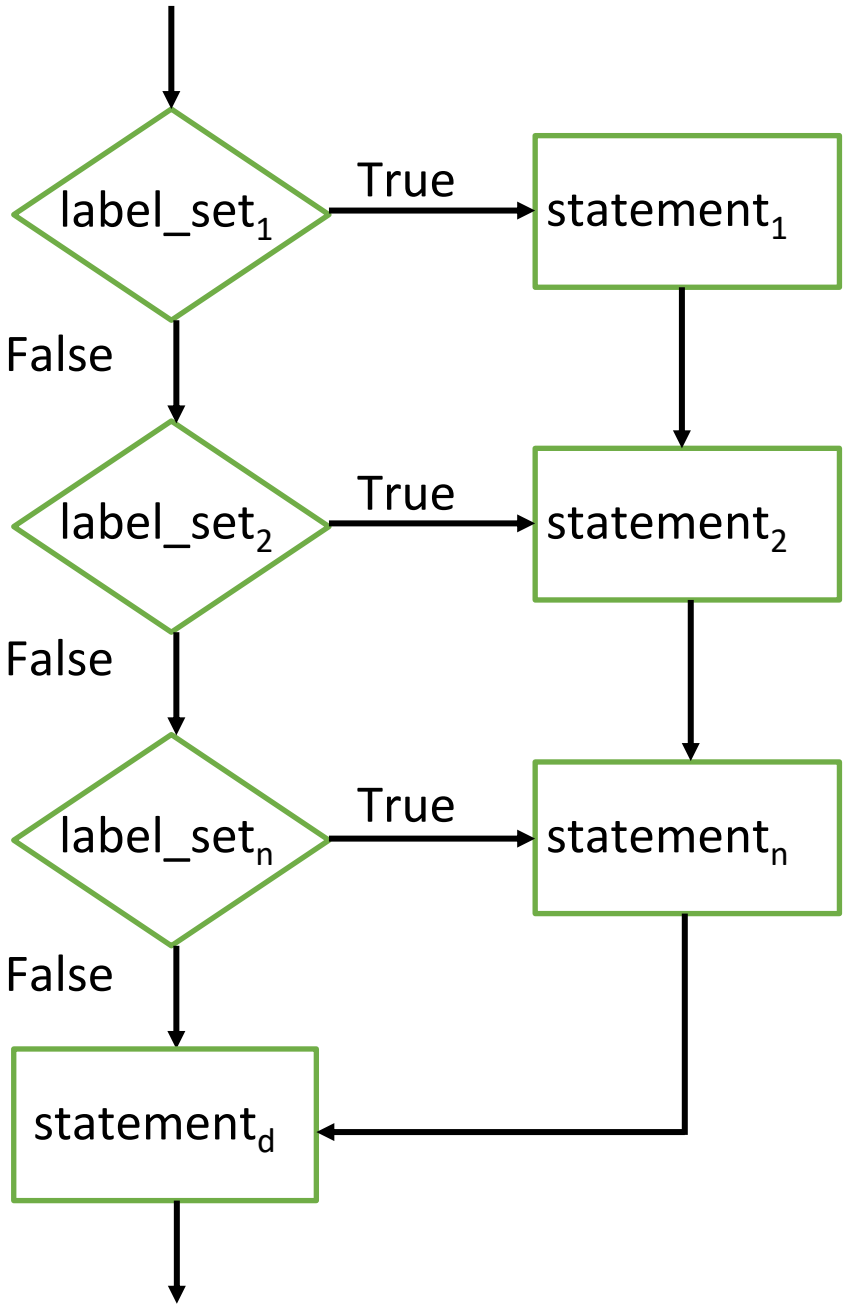
Switch Statement

- The **switch** statement may also be used in C to select one of several alternatives
- The switch statement is especially useful when the selection is based on the value of a single variable or of a simple expression
- Type int and char values may be used as case labels (the value of the expression), but strings and type double values cannot be used

```
switch(expression)
{
        case label_set1: statement1;
        case label_set2: statement2;
        ...
        case label_setn: statementn;
        [default: statementd;]
}
```

Switch Statement

```
switch(expression)
{
      case label_set1: statement1;
      case label_set2: statement2;
      ...
      case label_setn: statementn;
      [default: statementd;]
}
```



Switch Statement

```
switch(expression)
{
      case label_set1: statement1;
      case label_set2: statement2;
      ...
      case label_setn: statementn;
      [default: statementd;]
}
```

```
switch(ship)
{
    case 'B': printf("Battleship\n");
    case 'C': printf("Cruiser\n");
    case 'D': printf("Destroyer\n");
    default : printf("Unknown\n");
}
```

F:\switch.exe

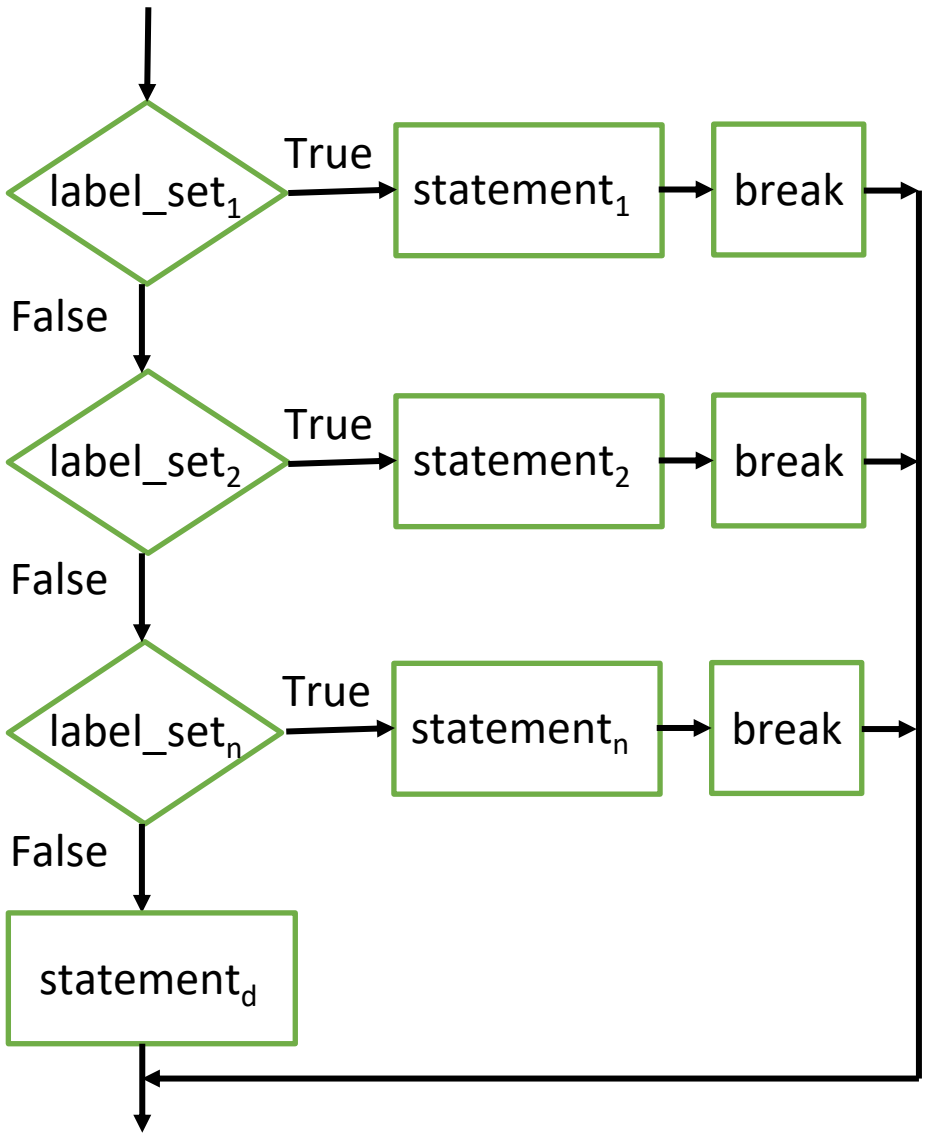
```
Ship: C
Cruiser
Destroyer
Unknown

Process returned 0 (0x0)
execution time : 1.344 s
Press any key to continue.
```

F:\switch.exe

```
Ship: F
Unknown

Process returned 0 (0x0)
execution time : 1.328 s
Press any key to continue.
```

Switch Statement

```
switch(expression)
{
        case label_set1:   statement1;
                           break;
        case label_set2:   statement2;
                           break;
        ...
        case label_setn:   statementn;
                           break;
        [default: statementd;]
}
```



Switch Statement

```
switch(expression)
{
        case label_set1:   statement1;
                           break;
        case label_set2:   statement2;
                           break;
        ...
        case label_setn:   statementn;
                           break;
        [default: statementd;]
}
```

F:\switch.exe

```
Ship: C
Cruiser

Process returned 0 (0x0)
execution time : 1.343 s
Press any key to continue.
```

F:\switch.exe

```
Ship: F
Unknown

Process returned 0 (0x0)
execution time : 1.282 s
Press any key to continue.
```

```
switch(ship)
{
    case 'B': printf("Battleship\n");
              break;
    case 'C': printf("Cruiser\n");
              break;
    case 'D': printf("Destroyer\n");
              break;
    default : printf("Unknown\n");
}
```

Practice 1

- What does the following code print?

```
#include <stdio.h>

int main()
{
    int noise;

    scanf("%d",&noise);
    if(noise <= 50) printf("Quiet\n");
    if(noise <= 70) printf("Intrusive\n");
    if(noise <= 90) printf("Annoying\n");
    if(noise <= 110) printf("Very Annoying\n");
    else printf("Uncomfortable\n");

    return 0;
}
```

Input :

```
80
```

Practice 2

- What does the following code print?

```
#include <stdio.h>

int main()
{
    int grade;

    scanf("%d",&grade);
    if(grade >= 55)
        printf("Passed\n");
    else
        printf("Failed\n");
        printf("You must take this course again\n");

    return 0;
}
```

Input :

```
85
```

Practice 3

- What does the following code print?

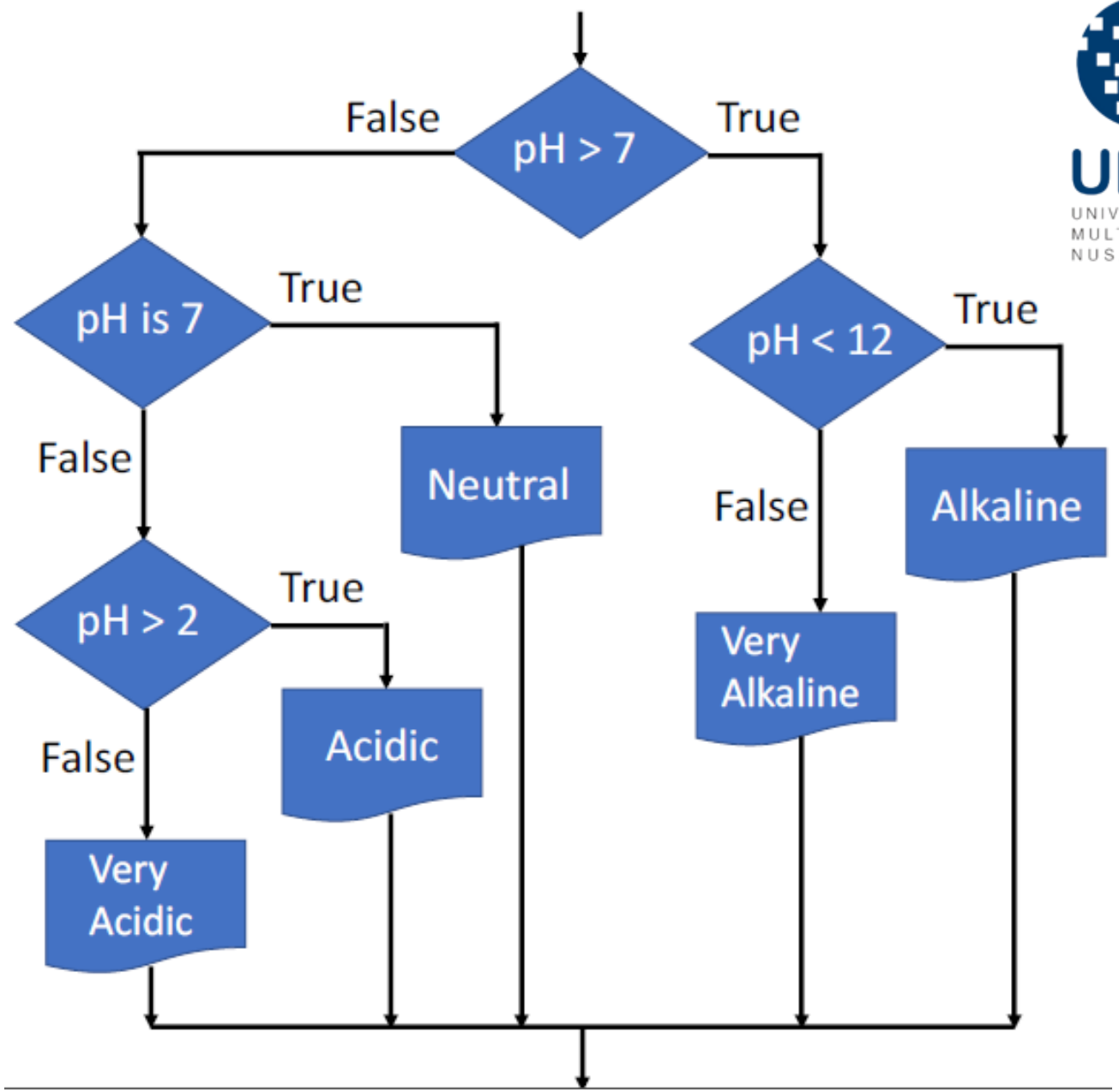
```c
#include <stdio.h>

int main()
{
    char color;

    color = getchar();
    switch(color)
    {
        case 'R': printf("Red\n");
        case 'G': printf("Green\n");
        case 'B': printf("Blue\n");
    }

    return 0;
}
```

Input :

```
R
```

Practice 4

- Write the **if statements** for the decision diagrammed in the accompanying flowchart.



Practice 5

- Write a switch statement that assigns to the variable **lumens** the expected brightness of a standard light bulb whose wattage has been stored in **watts**. Assign -1 to **lumens** if the value of **watts** is not in the table.

Watts	Brightness (in Lumens)
15	125
25	215
40	500
60	880
75	1000
100	1675

Practice 6

- Implement the following decision table using a **multiple-alternative if statement**. Assume that the wind speed is given as an integer. The screen dialogue should appear as follows:

Wind Speed (mph)	Category
Below 25	Not a strong wind
25 – 38	Strong wind
39 – 54	Gale
55 – 72	Whole gale
Above 72	Hurricane
Wind Speed (mph)	Category

```
Wind speed: 75

Category: Hurricane
```

```
Wind speed: 30

Category: Strong wind
```

NEXT WEEK'S OUTLINE

1. For
2. While…
3. Do…While…
4. Break
5. Continue

REFERENCES

- Hanly, Jeri R. and Koffman, Elliot B., 2013, Problem Solving and Program Design in C, Seventh Edition, Pearson Education, Inc.
- Deitel, Paul and Deitel, Harvey, 2016, C How to Program, Eighth Edition, Pearson Education, Inc.

Visi

Menjadi Program Studi Strata Satu Informatika **unggulan** yang menghasilkan lulusan **berwawasan internasional** yang **kompeten** di bidang Ilmu Komputer (*Computer Science*), **berjiwa wirausaha** dan **berbudi pekerti luhur**.

INFORMATIKA
UMN
UNIVERSITAS MULTIMEDIA NUSANTARA

Misi

1. Menyelenggarakan pembelajaran dengan teknologi dan kurikulum terbaik serta didukung tenaga pengajar profesional.
2. Melaksanakan kegiatan penelitian di bidang Informatika untuk memajukan ilmu dan teknologi Informatika.
3. Melaksanakan kegiatan pengabdian kepada masyarakat berbasis ilmu dan teknologi Informatika dalam rangka mengamalkan ilmu dan teknologi Informatika.