CS102/IT102
Computer Programming I

Lecture 8:
Selection (Part 1)

Bicol University College of Science
CSIT Department
1st Semester, 2023-2024

Topics

- Two-way selection
- The **`if`** statement
- The **`else`** statement
- Cascaded **`if`**
- Nested **`if`**
- Dangling **`else`**

Structured Programming

- Programs, written in 1950s and 1960s were a maze of complexity known as “spaghetti code”.
- In 1968, Edsger Dijkstra proposed that any program could be written with only three constructs or types of instructions:
  - **Sequences**: Built into C. Programs executed sequentially by default (sequential operations)
  - **Selection**: C has three types: `if`, `if…else`, and `switch` (conditional operations)
  - **Repetition**: C has three types: `while`, `do…while` and `for` (iterative operations)

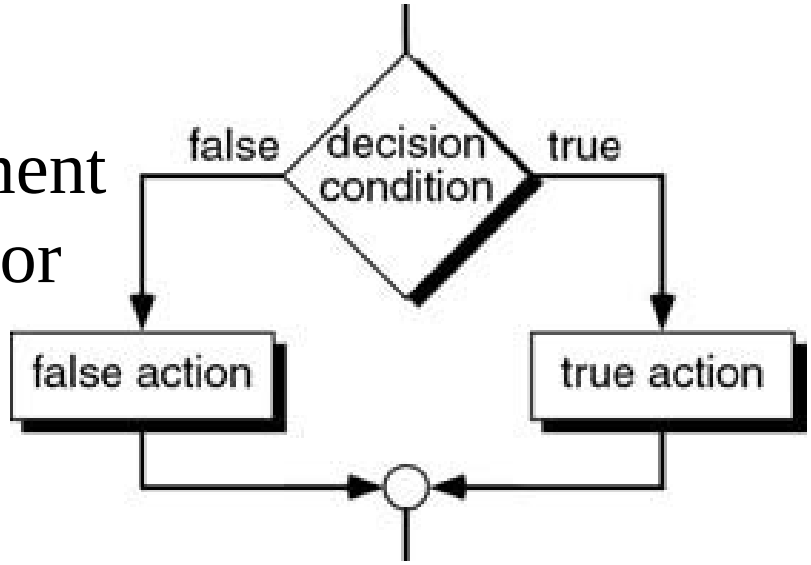
Structured Programming

- Today, virtually all programming languages offer structured programming capabilities.
- The second of the structured programming constructs is **selection**.
  - Selection allows you to choose between two or more alternatives: It allows you to make decisions.

Two-Way Selection

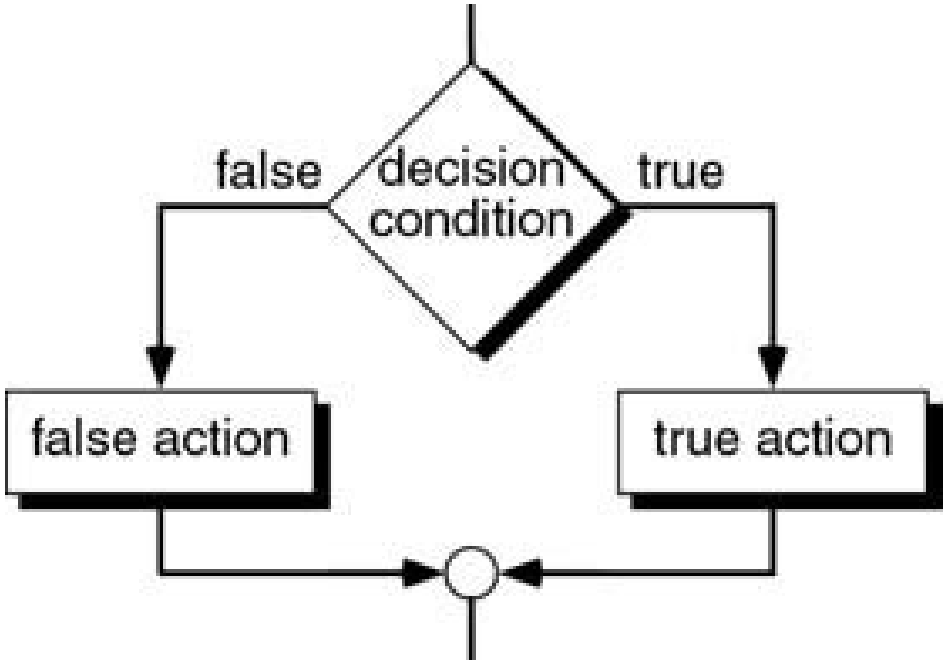
... the basic decision statement in the computer

- The decision is described to the computer as a conditional statement that can be answered either true or false.
  - If the answer is true, one or more action statements are executed.
  - If the answer is false, then a different action or a set of actions is executed.
  - Regardless of which set of action is executed, the program continues with the next statement after the selection



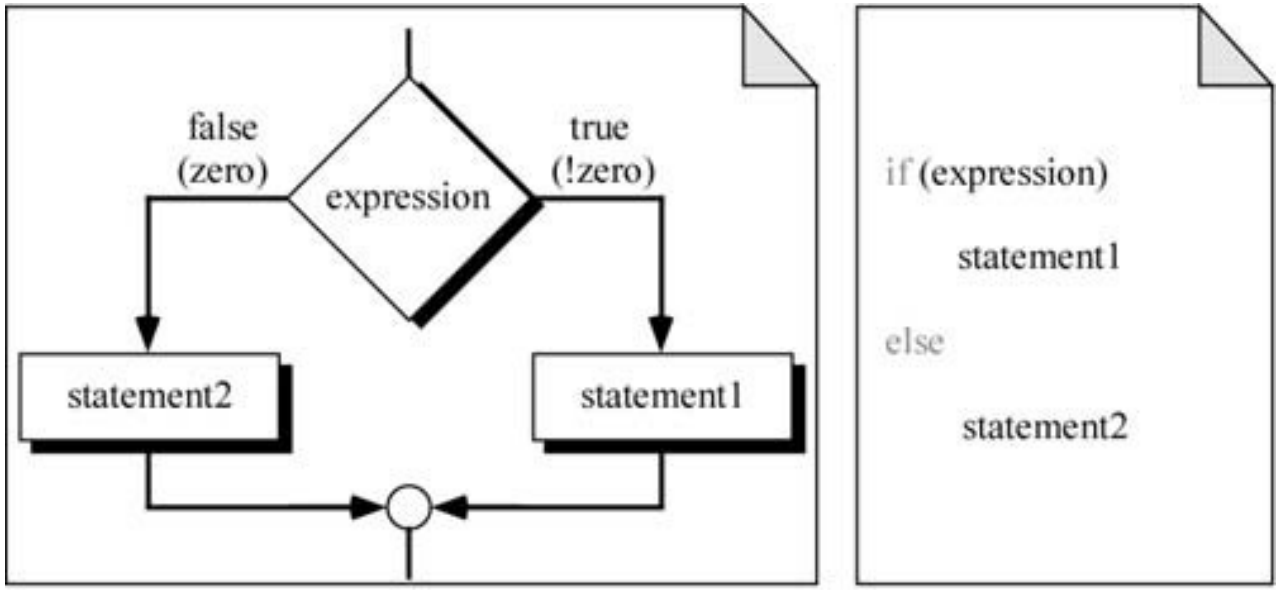
Two-Way Selection

- Diamond symbol (decision symbol)
  - Indicates decision is to be made
  - Contains an expression that can be true or false
  - Test the condition, follow appropriate path



The `if…else` statement

- C implements two-way selection with the **if…else** statement.
- An if…else statement is a composite statement used to make a decision between two alternatives.



(a) Logical Flow (b) Code

The `if` statement

- Determines whether a statement or block is executed.
- Implements the selection instructions within an algorithm.
- Decides what to do by evaluating a Boolean expression.
- If the expression is true (non-zero), the statement or block is executed.

```
if ( expression )
        statement
```

What is a statement?

- Statements are lines of instructions in our programs ending with a semicolon (;).
- A compound statement or block is a series of statements surrounded by braces.

E.g.

```
{
    number = number + 1;
    printf("%d\n", number);
}
```

- An empty statement is a single semicolon.

Example: oddnum.c

Read in a number, and print it if it is odd.

```
output “Enter an integer”
input number


if (number is odd)
then
{
   output the number
}
```

Example: oddnum.c

Read in a number, and print it if it is odd.

output “Enter an integer”
input number

if (number is odd)
then
{
  output the number
}

```
#include <stdio.h>

/* Read in a number, and echo it
   if it is odd. */

int main()
{

   return 0;
}
```

Example: oddnum.c

Read in a number, and print it if it is odd.

output “Enter an integer”
input number

if (number is odd)
then
{
  output the number
}

```
#include <stdio.h>

/* Read in a number, and echo it
   if it is odd. */

int main()
{
   int number;

   printf("Enter an integer: ");
   scanf("%d", &number);

   return 0;
}
```

Example: oddnum.c

Read in a number, and print it if it is odd.

output “Enter an integer”
input number

if (number is odd)
then
{
** output the number**
}

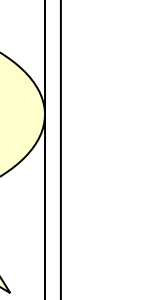
```
#include <stdio.h>

/* Read in a number, and echo it
   if it is odd. */

int main()
{
   int number;

   printf("Enter an integer: ");
   scanf("%d", &number);

   if (number % 2 != 0)
   {
      printf("%d\n", number);
   }

   return 0;
}
```

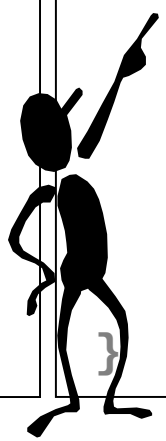
Example: oddnum.c

Read in a number, and print it if it is odd.

output "E
input nu

```
if (number is odd)
then
{
  output the number
}
```





```c
#include <stdio.h>

/* Read in a number, and echo it
   if it is odd. */

int main()
{
   int number;

   printf("Enter an integer: ");
   scanf("%d", &number);

   if (number % 2 != 0)
   {
      printf("%d\n", number);
   }

   return 0;
}
```

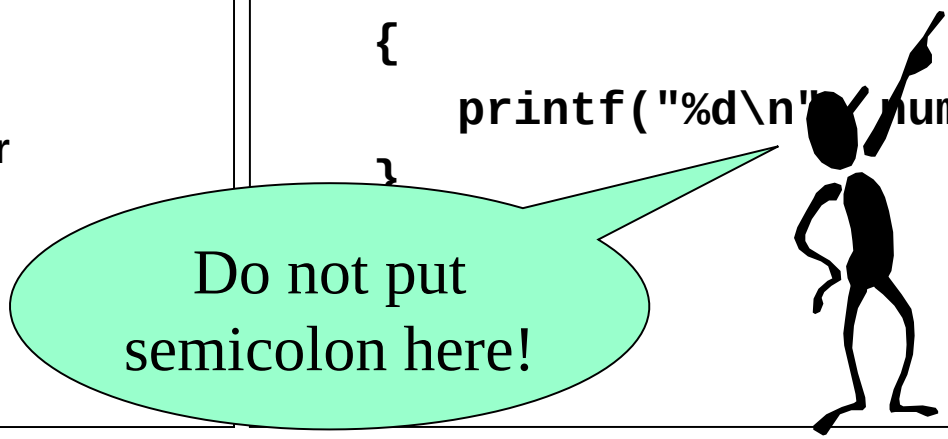
Example: oddnum.c

Read in a number, and print it if it is odd.

output “Enter an integer”
input number

if (number is odd)
then
{
** output the number**
}

```
#include <stdio.h>

/* Read in a number, and echo it
   if it is odd. */

int main()
{
   int number;

   printf("Enter an integer: ");
   scanf("%d", &number);

   if (number % 2 != 0)
   {
      printf("%d\n", number);
   }
```



Example: oddnum.c

Read in a number, and print it if it is odd.

output “Enter an integer”
input number

if (number is odd)
then
{
** output the number**
}

```
#include <stdio.h>

/* Read in a number, and echo it
   if it is odd. */

int main()
{
   int number;

   printf("Enter an integer: ");
   scanf("%d", &number);

   if (number % 2 != 0)
   {
      printf("%d\n", number);
   }

   return 0;
}
```

Notes on `if`

- Which of the following code fragments are equivalent?

A

```
if (number % 2 != 0)
{
   printf("%d", number);
}
printf(” is odd\n");
```

B

```
if (number % 2 != 0)
   printf("%d", number);
   printf(” is odd\n");
```

C

```
if (number % 2 != 0)
{
   printf("%d", number);
   printf(” is odd\n");
}
```

Notes on `if`

- Which of the following code fragments are equivalent?

A

```
if (number % 2 != 0)
{
   printf("%d", number);
}
printf(" is odd\n");
```

B

```
if (number % 2 != 0)
   printf("%d", number);
   printf(" is odd\n");
```

C

```
if (number % 2 != 0)
{
   printf("%d", number);
   printf(" is odd\n");
}
```

Notes on `if`

- Which of the following code fragments are equivalent?

A

```
if (number % 2 != 0)
{
   printf("%d", number);
}
printf(" is odd\n");
```

A Compound Statement

B

```
if (number % 2 != 0)
   printf("%d", number);
   printf(" is odd\n");
```

A Statement

C

```
if (number % 2 != 0)
{
   printf("%d", number);
   printf(" is odd\n");
}
```

Notes on `if`

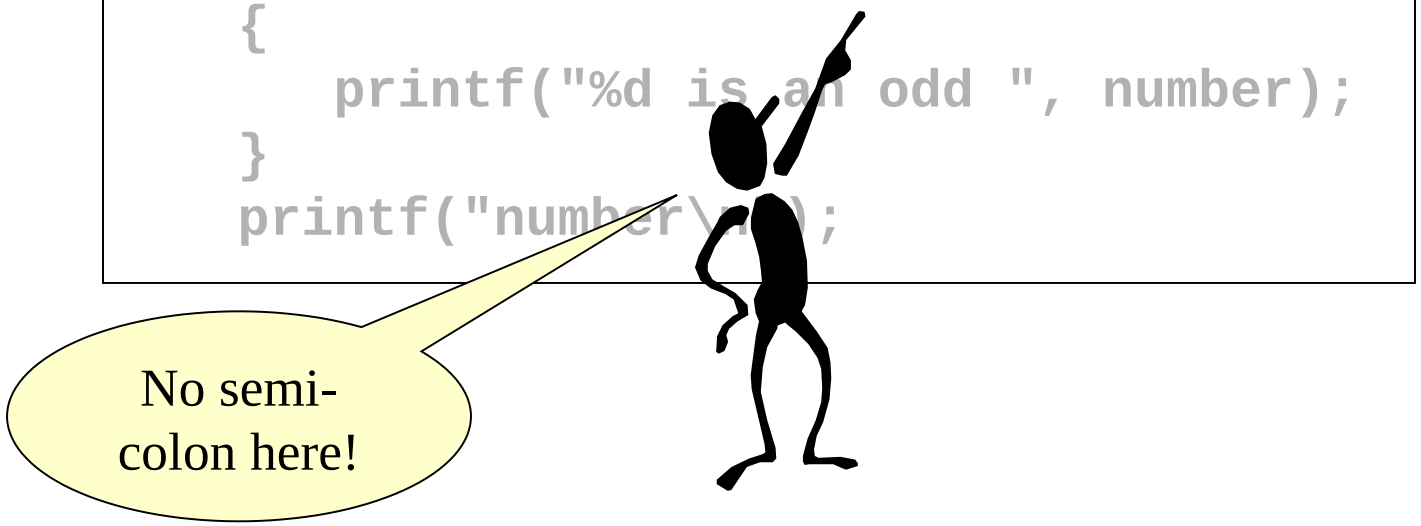
- Common mistake

```
if (number % 2 != 0);
{
   printf("%d is an odd ", number);
}
printf("number\n");
```

Notes on `if`

- Common mistake

```
if (number % 2 != 0);
{
   printf("%d is an odd ", number);
}
printf("number\n");
```



The semicolon is an empty statement.

Notes on `if`

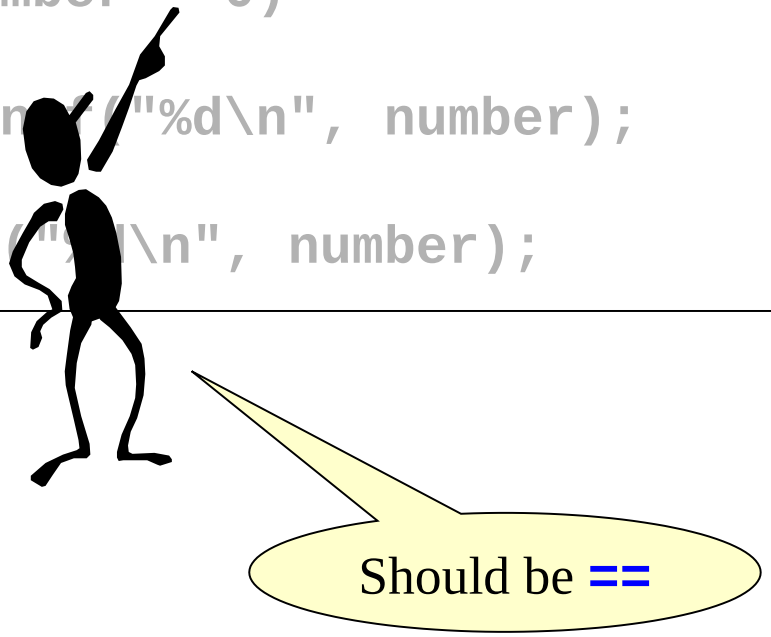
- Common mistake

```
if (number = 0)
{
   printf("%d\n", number);
}
printf("%d\n", number);
```

Notes on `if`

- Common mistake

```
if (number = 0)
{
    printf("%d\n", number);
}
printf("%d\n", number);
```



The `else` statement

- Can only occur after an **`if`** statement
- Is only executed when the **`if`** block does not execute

```
if ( expression )
        statement1
else
        statement2
```

Example: oddeven.c

Read in a number, and determine if it's odd or even.

```
output "Enter an integer"
input number

if (number is odd)
then
{
  output: number " is an odd
    number"
}
else
{
  output: number " is an even
    number"
}
```

```
#include <stdio.h>

/* Determine whether an input number
   is odd or even. */

main()
{
  int number;

  printf("Enter an integer: ");
  scanf("%d", &number);

  if (number % 2 != 0)
  {
    printf("%d is an odd number\n",
      number);
  }




}
```

Example: oddeven.c

Read in a number, and determine if it's odd or even.

output "Enter an integer"
input number

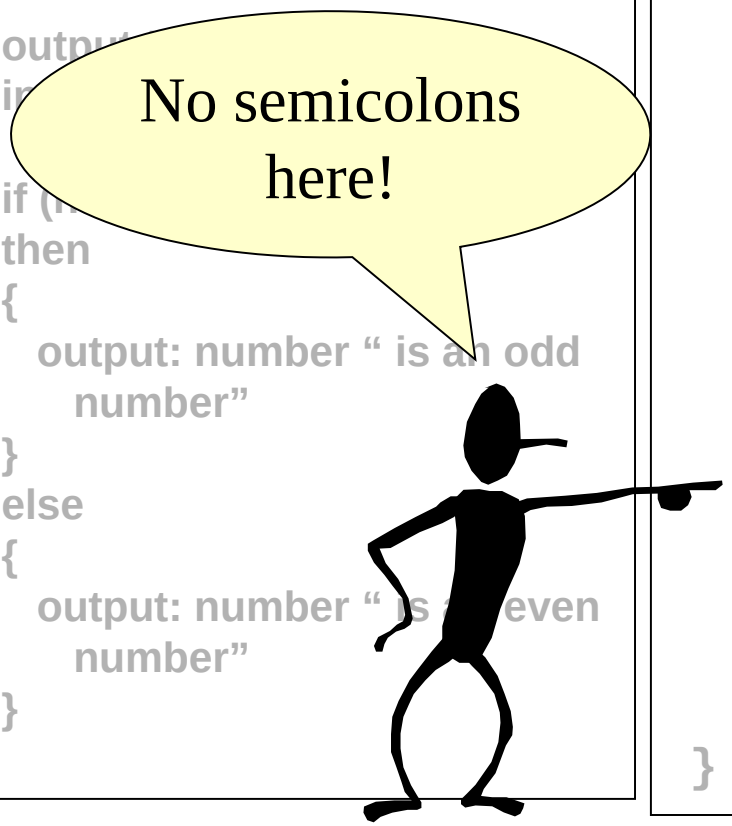
if (number is odd)
then
{
  output: number " is an odd number"
}
else
{
  output: number " is an even number"
}

```
#include <stdio.h>

/* Determine whether an input number
   is odd or even. */

main()
{
  int number;

  printf("Enter an integer: ");
  scanf("%d", &number);

  if (number % 2 != 0)
  {
    printf("%d is an odd number\n",
      number);
  }
  else
  {
    printf("%d is an even number\n",
      number);
  }
}
```

Example: oddeven.c

Read in a number, and determine if it's odd or even.

```
output
in
if (
then
{
  output: number " is an odd
    number"
}
else
{
  output: number " is an even
    number"
}
```




```
#include <stdio.h>

/* Determine whether an input number
   is odd or even. */

main()
{
  int number;

  printf("Enter an integer: ");
  scanf("%d", &number);

  if (number % 2 != 0)
  {
    printf("%d is an odd number\n",
      number);
  }
  else
  {
    printf("%d is an even number\n",
      number);
  }
}
```

Example: oddeven.c

Read in a number, and determine if it's odd or even.

output "Enter an integer"
input number

if (number is odd)
then
{
output: number " is an odd number"
}
else
{
output: number " is an even number"
}

```
#include <stdio.h>

/* Determine whether an input number
   is odd or even. */

main()
{
  int number;

  printf("Enter an integer: ");
  scanf("%d", &number);

  if (number % 2 != 0)
  {
    printf("%d is an odd number\n",
      number);
  }
  else
  {
    printf("%d is an even number\n",
      number);
  }
}
```

Cascaded `if` statement

- Multiple alternative blocks each with a Boolean expression.
- First expression which evaluates to true causes execution of the associated block.
- At most only one block will be executed.

Example: months.c

Determine the number of days in a given month:

30 days hath September,
April, June and November.
All the rest hath 31,
Excepting February alone,
Which hath 28 days clear,
And 29 in each leap year.

```
output “Enter an integer”
input month

if (month is September,
            or April,
            or June,
            or November)
then
{
  output “30 days”
}
else if (month is February)
{
  output “28 or 29 days”
}
else
{
  output “31 days”
}
```

Example: months.c

```
#include <stdio.h>

/*************************\

Determine the number of days
   in a given month:

30 days hath September,
April, June and November;
All the rest have 31,
Excepting February alone,
And that has 28 days clear
And 29 in each leap year.
\*************************/

const int September = 9;
const int April = 4;
const int June = 6;
const int November = 11;
const int February = 2;
```

```
int main()
{




  return 0;
}
```

Example: months.c

```
#include <stdio.h>

/*************************\

Determine the number of days
   in a given month:

30 days hath September,
April, June and November;
All the rest have 31,
Excepting February alone,
And that has 28 days clear
And 29 in each leap year.
\*************************/

const int September = 9;
const int April = 4;
const int June = 6;
const int November = 11;
const int February = 2;
```

```
int main()
{
  int month;
  printf("Enter number of month: ");
  scanf("%d", &month);

  return 0;
}
```

Example: months.c

```
#include <stdio.h>

/*************************\

Determine the number of days
   in a given month:

30 days hath September,
April, June and November;
All the rest have 31,
Excepting February alone,
And that has 28 days clear
And 29 in each leap year.
\*************************/

const int September = 9;
const int April = 4;
const int June = 6;
const int November = 11;
const int February = 2;
```

```
int main()
{
  int month;
  printf("Enter number of month: ");
  scanf("%d", &month);

  if (month==September ||
      month==April ||
      month==June ||
      month==November )
  {
    printf("30 days\n");
  }

  return 0;
}
```

Example: months.c

```
#include <stdio.h>

/*************************\

Determine the number of days
   in a given month:

30 days hath September,
April, June and November
All the rest have 31,
Excepting February
And that has 28 d
And 29 in each
\************
```

```
int main()
{
  int month;
  printf("Enter number of month: ");
  scanf("%d", &month);

  if (month==September ||
      month==April ||
      month==June ||
      month==November )
  {
    printf("30 days\n");
  }
```

Common mistake:

```
if (month==September || April || June || November )
```

Example: months.c

```
#include <stdio.h>

/*************************\

Determine the number of days
   in a given month:

30 days hath September,
April, June and November;
All the rest have 31,
Excepting February alone,
And that has 28 days clear
And 29 in each leap year.
\*************************/

const int September = 9;
const int April = 4;
const int June = 6;
const int November = 11;
const int February = 2;
```

```
int main()
{
  int month;
  printf("Enter number of month: ");
  scanf("%d", &month);

  if (month==September ||
      month==April ||
      month==June ||
      month==November )
  {
    printf("30 days\n");
  }
  else if (month==February)
  {
    printf("28 or 29 days\n");
  }




  return 0;
}
```

Example: months.c

```
#include <stdio.h>

/*************************\

Determine the number of days
   in a given month:

30 days hath September,
April, June and November;
All the rest have 31,
Excepting February alone,
And that has 28 days clear
And 29 in each leap year.
\*************************/

const int September = 9;
const int April = 4;
const int June = 6;
const int November = 11;
const int February = 2;
```

```
int main()
{
  int month;
  printf("Enter number of month: ");
  scanf("%d", &month);

  if (month==September ||
      month==April ||
      month==June ||
      month==November )
  {
    printf("30 days\n");
  }
  else if (month==February)
  {
    printf("28 or 29 days\n");
  }
  else
  {
    printf("31 days\n");
  }
  return 0;
}
```

Example: months.c

```
#include <stdio.h>

/*************************\

Determine the number of days
   in a given month:

\*********************

const int September = 9;
const int April = 4;
const int June = 6;
const int November = 11;
const int February = 2;
```

```
int main()
{
  int month;
  printf("Enter number of month: ");
  scanf("%d", &month);

  if (month==September ||
      month==April ||
      month==June ||
      month==November )
  {
    printf("30 days\n");
  }
  else if (month==February)
  {
    printf("28 or 29 days\n");
  }
  else
  {
    printf("31 days\n");
  }
  return 0;
}
```

"Default" block.

Example: months.c

```
#include <stdio.h>

/*************************\

Determine the number of days
   in a given month:

30 days hath September,
April, June and November;
All the rest have 31,
Excepting February alone,
And that has 28 days clear
And 29 in each leap year.
\*************************/

const int September = 9;
const int April = 4;
const int June = 6;
const int November = 11;
const int February = 2;
```

```
int main()
{
  int month;
  printf("Enter number of month: ");
  scanf("%d", &month);

  if (month==September ||
      month==April ||
      month==June ||
      month==November )
  {
    printf("30 days\n");
  }
  else if (month==February)
  {
    printf("28 or 29 days\n");
  }
  else
  {
    printf("31 days\n");
  }
  return 0;
}
```

Notes on Cascaded `if`

Q:

What is the output if:

- letter is equal to ‘b’
- letter is equal to ‘z’
- letter is equal to ‘A’
- letter is equal to ‘X’

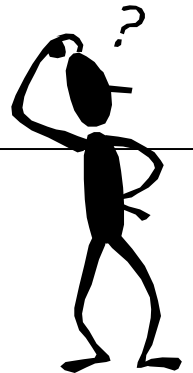
```
if (letter >= 'a')
{
   printf("S1\n");
}
else if (letter <= 'z')
{
   printf("S2\n");
}
else if (letter >= 'A')
{
   printf("S3\n");
}
else if (letter <= 'Z')
{
   printf("S4\n");
}
```

More Examples

```
if (ch >= 'a' && ch <= 'z')
{
   printf("%c is in lower case.\n", ch);
}
else if (ch >= 'A' && ch <= 'Z')
{
   printf("%c is in upper case.\n". ch);
}
else if (ch >= '0' && ch <= '9')
{
   printf("%c is a digit with value %d.\n", ch, ch - '0');
}
```

More Examples

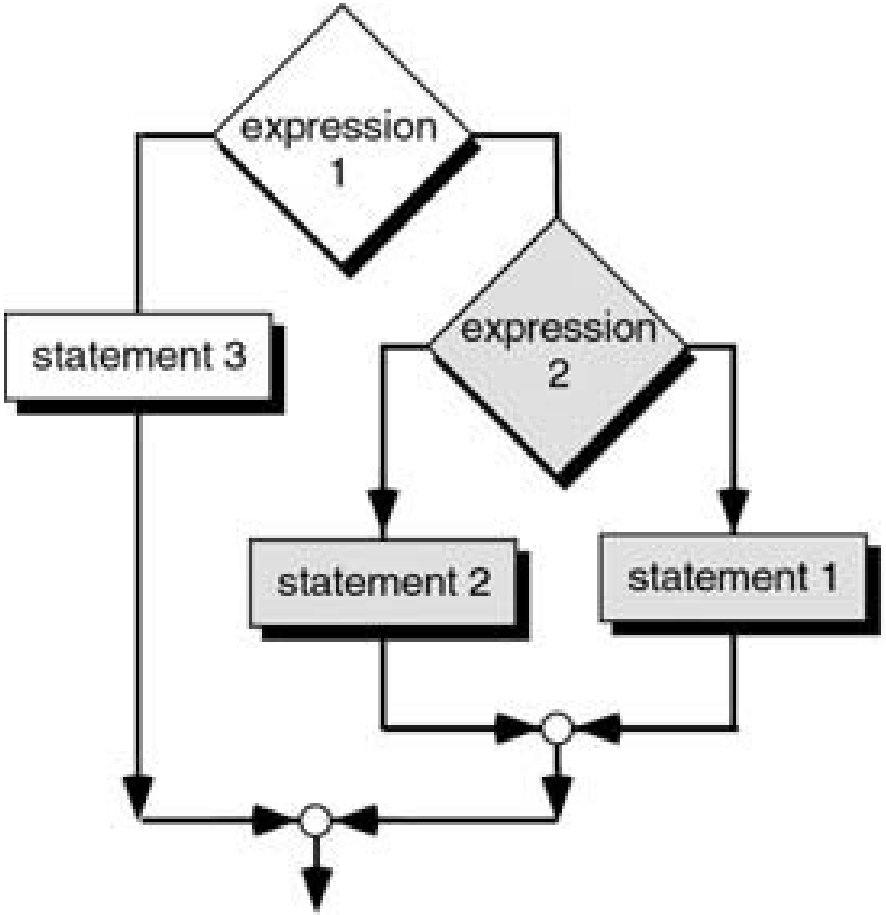
```
if (ch >= 'a' && ch <= 'z')
{
   printf("%c is in lower case.\n", ch);
}
else if (ch >= 'A' && ch <= 'Z')
{
   printf("%c is in upper case.\n". ch);
}
else if (ch >= '0' && ch <= '9')
{
   printf("%c is a digit with value %d.\n", ch, ch - '0');
}
```



Nested `if` statement

- For the `if…else`, the statements may be any statement, including another `if…else`.
- When if…else is included within an `if…else`, it is known as a **nested if statement**.
- There is no limit to how many levels can be nested, but if there are more than three, they can become difficult to read.

Nested `if` statement



(a) Logic flow

```
if (expression 1)
    if (expression 2)
        statement 1
    else
        statement 2
else
    statement 3
```

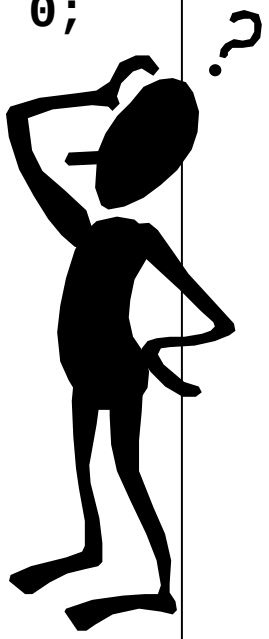
(b) Code

Example: Nested `if`

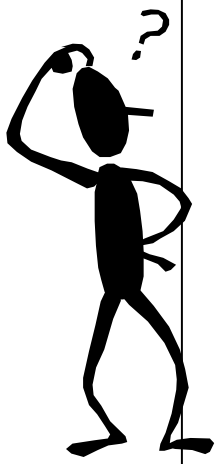
```
if ( coldWeather )
{
   wearJumper = 1;

   wearRaincoat = wearJacket = wearThermal = 0;

   if ( raining )
      wearRaincoat = 1;
   else
      wearJacket = 1;

   if ( belowZero )
   {
      wearThermal = 1;
   }
}
```

Example: Nested `if`

```
if ( coldWeather )
{
   wearJumper = 1;

   wearRaincoat = wearJacket = wearThermal = 0;

   if ( raining )
      wearRaincoat = 1;
   else
      wearJacket = 1;

   if ( belowZero )
   {
      wearThermal = 1;
   }
}
```



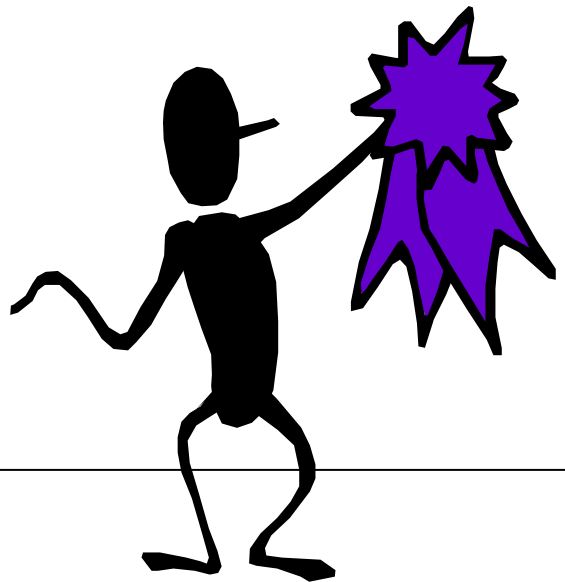
Example: Nested `if`



```
if ( coldWeather )
{
   wearJumper = 1;

   wearRaincoat = wearJacket = wearThermal = 0;

   if ( raining )
      wearRaincoat = 1;
   else
      wearJacket = 1;

   if ( belowZero )
   {
      wearThermal = 1;
   }
}
```

Example: Nested **`if`**

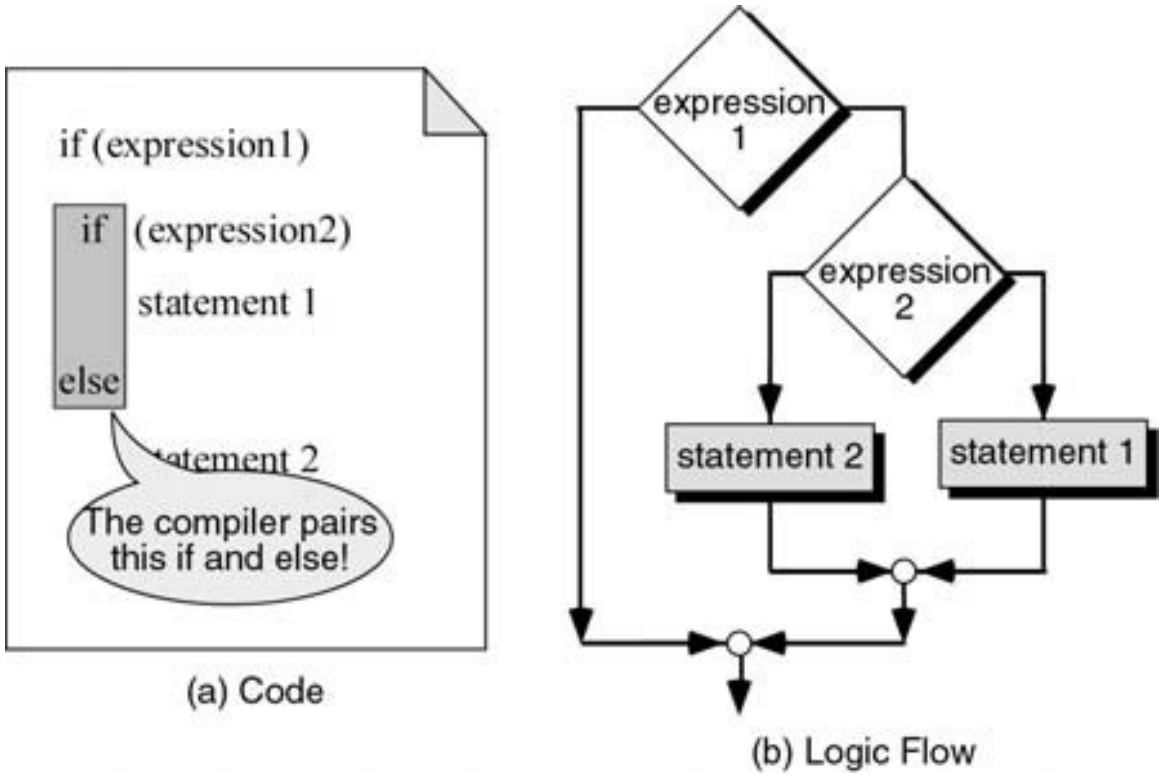
```
if ( coldWeather )
{
   wearJumper = 1;

   wearRaincoat = wearJacket = wearThermal = 0;

   if ( raining )
      wearRaincoat = 1;
   else
      wearJacket = 1;

   if ( belowZero )
   {
      wearThermal = 1;
   }
}
```



Dangling `else` Problem

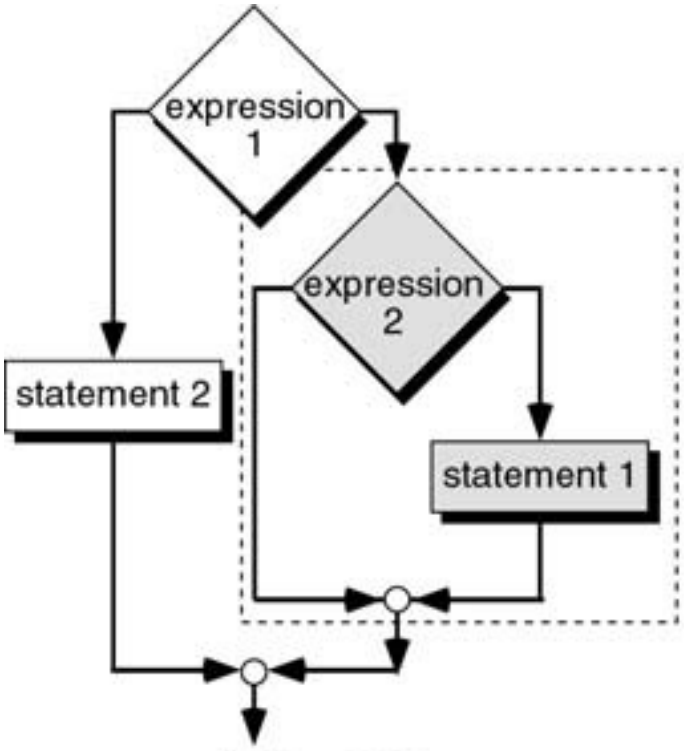
... is created when there is no matching else for every if.

- C's solution to is a simple rule: **Always pair an else to the most recent unpaired if in the current block.**
- This rule may result in some if statement being left unpaired.
- Have to ensure that the resulting code is what you need.

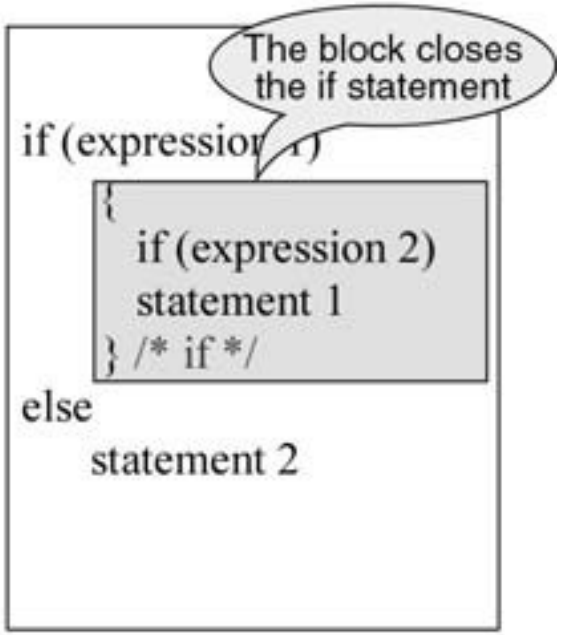


Dangling `else` Solution

- is a compound statement, i.e. enclose the true actions in braces to make the second if a compound statement.
- Since the closing brace completes the body of the compound statement, the else is automatically paired with the correct if.



(a) Logic Flow



(b) Code

Summary

- Two-way selection
- The **`if`** statement
- The **`else`** statement
- Cascaded **`if`**
- Nested **`if`**
- Dangling **`else`**
- Common mistakes

Readings

This Lecture:

- King: Section 5.2
- D&D: Sections 3.7, 4.1 – 4.6, 4.8 – 4.11
- Kernighan & Ritchie: 3.1 – 3.3

Readings: D&D (2/e or 3/e):
Sections 3.4 to 3.6 and 4.10 to 4.11