

Selection in C++

CS1A


- ☀ Flowcharts to code
- ☀ IF-THEN
 - ☀ NESTED IF-THEN statements
- ☀ IF-THEN-ELSE
 - ☀ NESTED IF-THEN-ELSE

(c) Michele Rousseau

Selection

1

3 Basic Control / Logic Structures

- ☀ Sequence 
 - Instructions are executed **one after another** in the **order** they appear in the program
 - Until another control structure takes precedence
- ☀ Selection
 - Based on some **condition**, either **one part** of the program is executed **or another part** is executed
 - The program chooses which part to execute based on the condition
- ☀ Repetition
 - Part of the code is **executed over and over** (repeated)
 - This can be for a set number of times or until a condition is met

Today we will focus on **Selection**.
Don't worry → We'll get to the others later on.

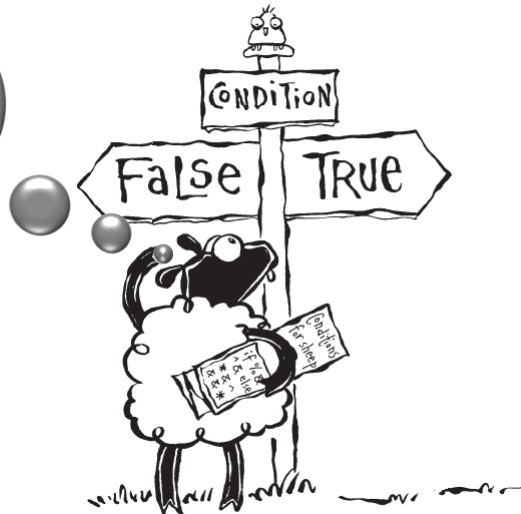
(c) Michele Rousseau

Selection

2

Selection Structures

What if I only want some instructions to run some of the time?



(c) Michele Rousseau

Selection

3

Selection Structures

Selection

→ Choosing between two or more alternative actions

- Run certain instructions based on some **condition**
- Conditions are based on **Boolean Expressions**
 - An expression that evaluates to 1 of 2 possibilities
 - Either **True** or **False**
- The computer evaluates a **Boolean Expression** and determines which instructions to execute based on the result
- **Boolean expressions** are formed using **relational operators**

(c) Michele Rousseau

Selection

4

Relational Operators

We use **Relational Operators** to form Boolean Expressions

==	Equal
<	Less than
>	Greater than
<=	Less than or equal
>=	Greater than or equal
!=	Not Equal

NOTE: this is not the same as =,
= ← is an assignment

We use **Relational Operators** to compare values in Selection Statements

→ These will return a True (1) or False (0) value.

(c) Michele Rousseau

Selection

5

Examples of Boolean Expressions

What will be the result of these Boolean functions?

5 == 5

'a' < 'c'

4 + 3 > 10

10 != 20

6 <= 6

5 >= 9

'A' < 'Z'

'a' < 'Z'

If you compare characters using **relational operators**,
→ It compares the **ASCII values**
(in this case 'a' has a greater ASCII value than 'Z')

(c) Michele Rousseau

Selection

6

ASCII Chart for
Printing
Characters

Char	Decimal Value
SP	32
!	33
"	34
#	35
\$	36
%	37
&	38
'	39
(40
)	41
*	42
+	43
,	44
-	45
.	46
/	47
0	48
1	49
2	50
3	51
4	52
5	53
6	54
7	55

Char	Dec
8	56
9	57
:	58
;	59
<	60
=	61
>	62
?	63
@	64
A	65
B	66
C	67
D	68
E	69
F	70
G	71
H	72
I	73
J	74
K	75
L	76
M	77
N	78
O	79

Char	Dec
P	80
Q	81
R	82
S	83
T	84
U	85
V	86
W	87
X	88
Y	89
Z	90
[91
\	92
]	93
^	94
_	95
`	96
a	97
b	98
c	99
d	100
e	101
f	102
g	103

Char	Dec
h	104
i	105
j	106
k	107
l	108
m	109
n	110
o	111
p	112
q	113
r	114
s	115
t	116
u	117
v	118
w	119
x	120
y	121
z	122
{	123
	124
}	125
~	126
DEL	127

(c) Michele Rousseau
Selection
7

3-types of Selection Statements

- One-way Decisions
 - If-Then Statements
 - If the condition is **true** then **execute some instructions**
 - If the condition is **false** → **don't do anything special**
- Two-way Decisions
 - If-Then-Else Statements
 - If the condition is **true** then **execute some instructions**
 - else (the condition is **false**) **execute another set of instructions**
- Multi-way Decisions
 - Nested If-Then or Nested If-Then-Else Statements
 - Many options...

If Statements

- If statements take different forms
 - For now we will focus on the 2 basic forms
 - ▣ If-Then
 - ▣ If-Then-Else
 - Both of these statements can be nested
- A simple “if-then statement” is a one-way stmt
 - One-way decisions
 - ▣ If a condition is **true** → execute some special instructions

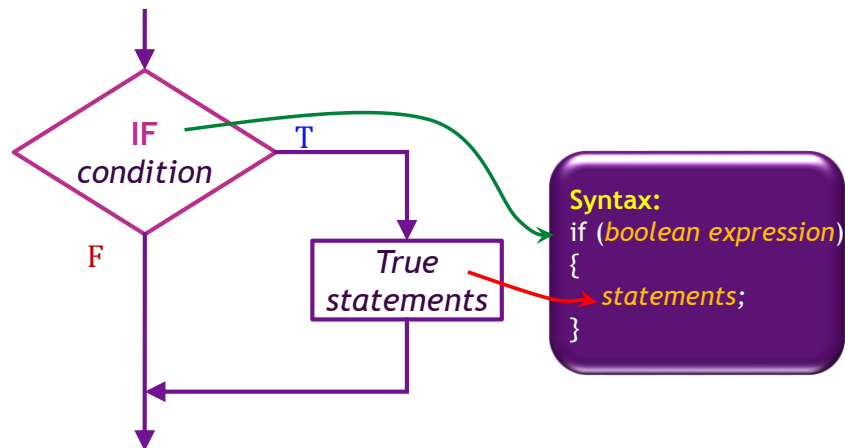
Syntax:
if (*boolean expression*)
{
 statements;
}

(c) Michele Rousseau

Selection

9

If-Then Statement



1. The Boolean expression is evaluated
2. If it evaluates to **TRUE**, then the **True Instructions** are executed
3. If it evaluates to **FALSE** the statement is ignored and the program continues with the next executable statement

(c) Michele Rousseau

Selection

10

If – Then Statement

- If some condition is true then we execute some set of instructions
- Otherwise we don't do anything special

IF Joe is 18 or older **THEN**
tell him he can vote

Can I vote?

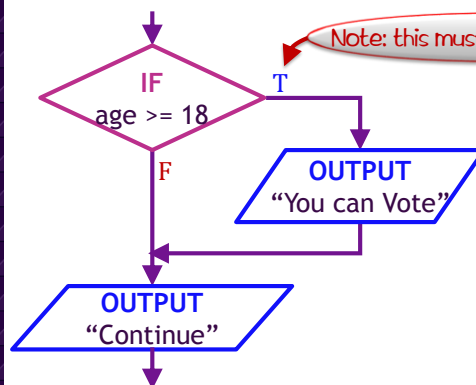


(c) Michele Rousseau

Selection

11

Exercise: If-Then Example



Syntax:
if (*boolean expression*)
{
 statements;
}

What would the output be for...

age = 18?

age = 16?

(c) Michele Rousseau

Selection

12

Nesting If-Then Statements

- If we need to check if more than one condition is true to execute some instructions
- Otherwise we don't do anything special

IF Joe is over 18 **THEN**
IF Joe is registered **THEN**
tell him he can vote.

Okay, I am over 18
now can I vote?



(c) Michele Rousseau

Selection

13

Nested If Statements

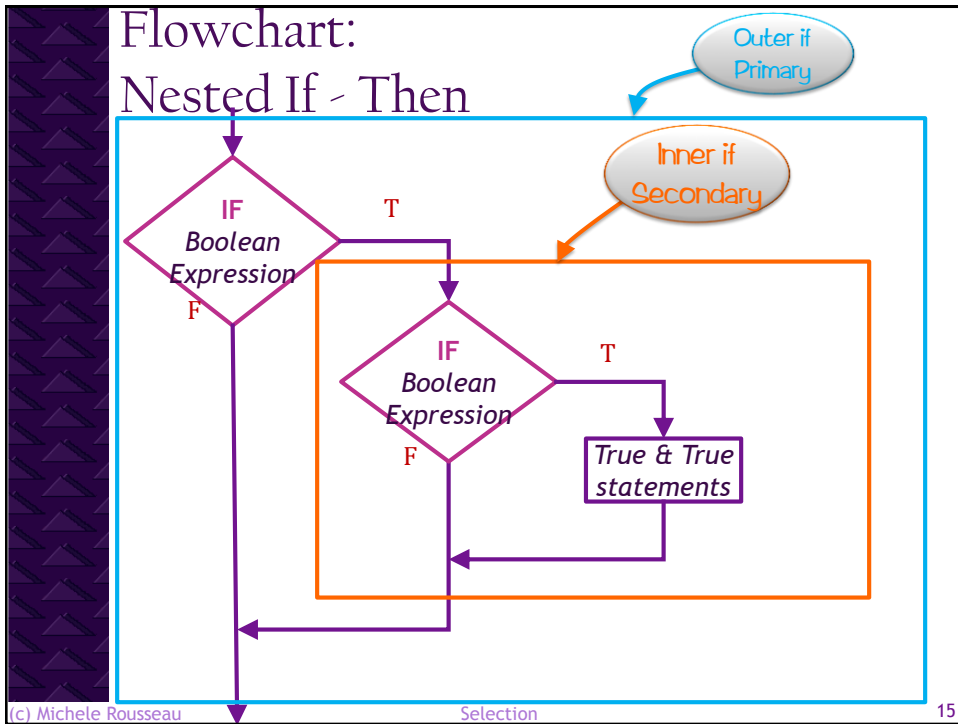
- Nested Selection Structure
 - Selection structure within another selection structure
 - Used when more than one decision must be made before an appropriate action can be carried out
- Primary Decision
 - always made by the outer selection structure
- Secondary Decision
 - Always made in the inner (or nested) selection structure
- Nested If-Then statement
 - an If-Then statement that **contains another** If-Then statement (within the statement section)

(c) Michele Rousseau

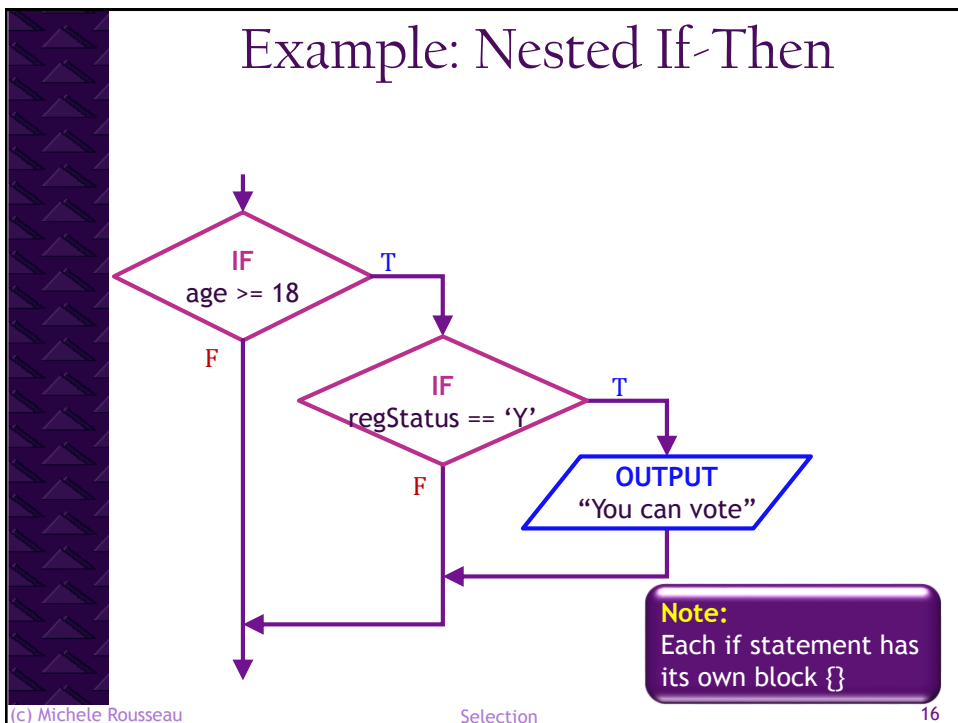
Selection

14

Flowchart: Nested If - Then



Example: Nested If-Then



If-Then-Else Statements

- If some condition is true then we execute some set of instructions
- else (when the condition is false) we execute a different set of instructions

IF Joe is 18 or older **THEN**
tell him he can vote,
ELSE
tell him he can't vote

Hmmm... It didn't
tell me that I can vote,
but it didn't say I can't
vote either



(c) Michele Rousseau

Selection

17

If-Then-Else Statements

Two-way Decisions

- Either execute one set of instructions or another
- Based on a Boolean expression

If the condition is true then

- Execute one set of instructions

Else

- Execute another set of instructions

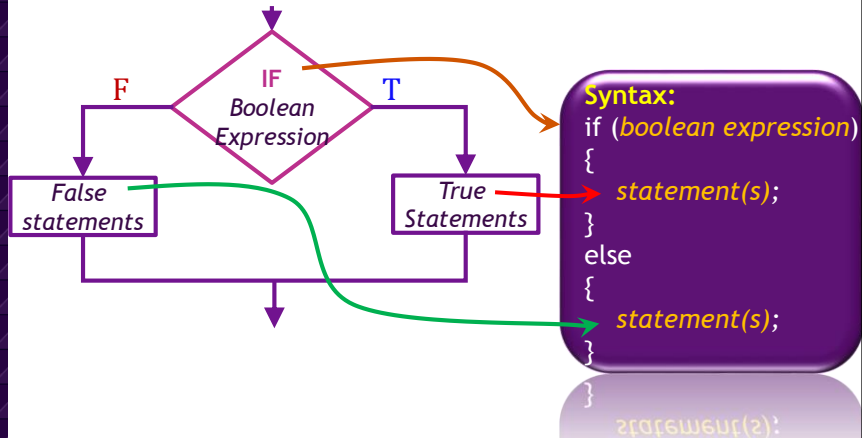
Syntax:

```
if (boolean expression)
{
    statement(s);
}
else
{
    statement(s);
}
```

(c) Michele Rousseau

18

Flowchart for If-Then-Else Stmt



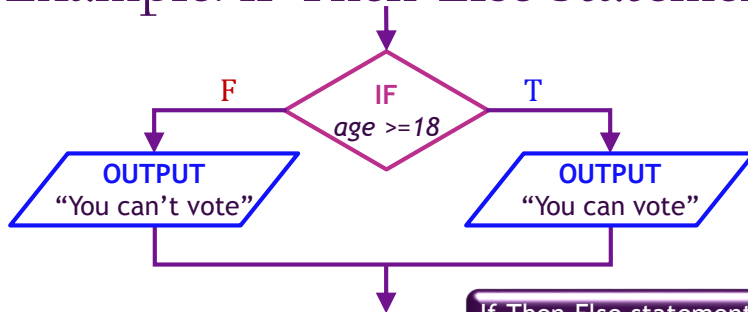
1. The Boolean expression is evaluated
2. If it evaluates to **TRUE**, then **statement1** is executed
3. If it evaluates to **FALSE**, then **statement2** is executed

(c) Michele Rousseau

Selection

19

Example: If-Then-Else Statement



Syntax:

```

if (boolean expression)
{
    statement(s);
}
else
{
    statement(s);
}
  
```

If-Then-Else statements
can also be nested

(c) Michele Rousseau

Selection

20

Nesting If-Then-Else Statements

- If we need to check if more than one condition is true to execute some instructions
- Otherwise we don't do anything special

IF Joe is over 18 **THEN**
 IF Joe is registered **THEN**
 Tell him he can vote
 ELSE
 Tell him he needs to register
ELSE
 Tell him he is too young to vote

So many requirements

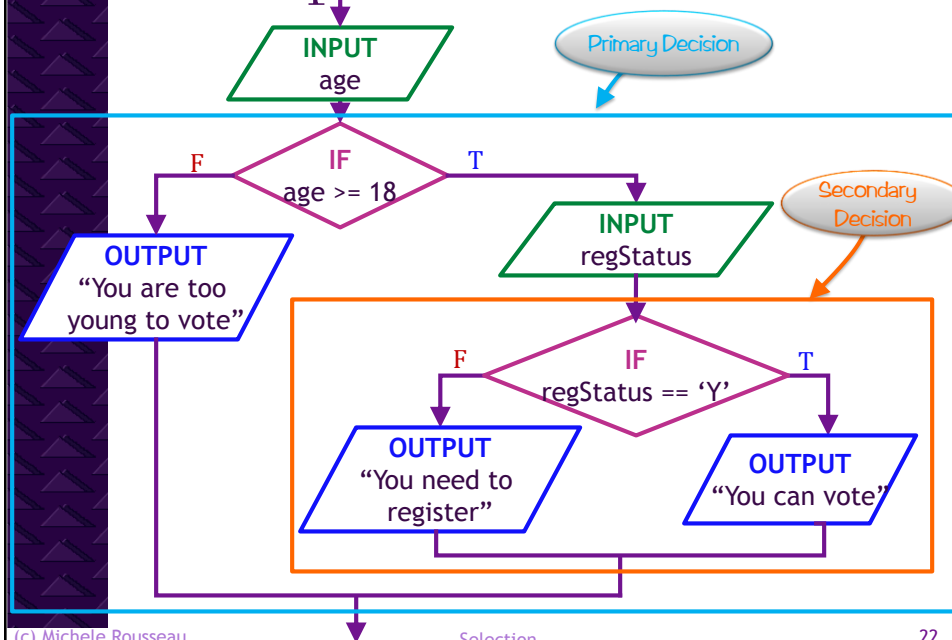


(c) Michele Rousseau

Selection

21

Example: Nested If-Then-Else



(c) Michele Rousseau

Selection

22

Example: Nested If-Then-Else

(c) Michele Rousseau

Selection

23

Nesting IF statements

- We could have nested on the false side of the if-then-else
- We could also nest more than one time
 - Given the syntax...
 - We can put any statement in between the brackets { }
 - An *if statement* is just another statement so it too can go between the brackets

Syntax:

```
if (boolean expression)  
{  
    statement(s);  
}  
else  
{  
    statement(s);  
}
```

(c) Michele Rousseau

Selection

24

Common Errors in Selection Structures

◦ Syntax errors:

- Forgetting the parenthesis
 - ▣ Eg. `if (age >= 18)` → NOT `if age >= 18`
- Putting a “;” at the end of the first line
 - ▣ Eg. `if (age >= 18)` → NOT `if (age >= 18);`
 - ▣ The statement is correct it just won't do anything

(c) Michele Rousseau

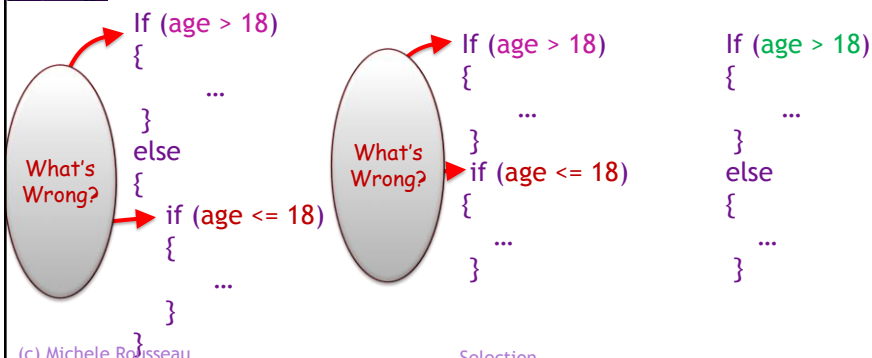
Selection

25

Common Logic Errors in If Structures

◦ Logic errors are commonly made as a result of the following mistakes:

- Reversing the primary and secondary decisions
- Redundancy
 - ▣ Using an unnecessary nested selection structure
- Using if-then instead of if-then-else



(c) Michele Rousseau

Selection

26

Comparing c-strings

Cstrings are stored in an array

- Remember an array is a contiguous area of storage where each element has the same data type
- cstrings are an array of characters
- When you access an array you are working with the address of the array → Not the value in the address
- When you make the following comparison:
`if(stringOne == stringTwo)`

you are comparing the addresses → not the values
→ the addresses will never be the same

strcmp function

```
strcmp(c-string1, c-string2);
```

- Compares the contents of string1 and string 2
- Returns:

Return value	if ASCII VALUES are such that
0	string1 == string2
Integer < 0	string1 < string2
Integer > 0	string1 > string2

NOTE: String comparisons should be of the same size
→ can't do
`char str1[8];`
`char str2[10];`
If `(str1 == str2)` <- can't compare these

Example

```
char stringOne[10];
char stringTwo[10];

cout << "Enter the first string: ";
cin  >> stringOne;

cout << "Enter the second string: ";
cin  >> stringTwo;

if(strcmp(stringOne, stringTwo) == 0)
{
    cout << "The strings are the same";
}
```

(c) Michele Rousseau

Selection

29

Assigning C-Strings

Similarly, we can't directly assign one c-string into another

For example:

```
char name1[30];
char name2[30];
```

We can't do this: `name1 = "Joe";`

Instead we can use:

```
strncpy(toC-string, fromC-string2 , sizeOfToC-string );
```

```
strncpy(name1, "Joe", 30);
cout << name1;
```

This would output: **Joe**

```
strncpy(name2, "Mo", 30);
strncpy(name1, name2, 30);
cout << name1;
```

This would output: **Mo**

NOTE: You need to `#include <cstring>` to use `strncpy`

(c) Michele Rousseau

Selection

30