If Statements

Principles of Computer Programming I

Spring/Fall 20XX

AUGUSTA UNIVERSITY

Outline

- If and if-else statements
- Nested if statements
- If-else-if statements

Conditional Execution

- `if` statement: Execute a code block **only** if a condition is true

```
Console.WriteLine("Enter your age");
int age = int.Parse(Console.ReadLine());
if(age >= 18)
{
  Console.WriteLine("You can vote!");
}
Console.WriteLine("Goodbye");
```

Condition

Code block marked by {}

```
Enter your age
20
You can vote!
Goodbye
```

```
Enter your age
17
Goodbye
```

If Statement Rules

- Expression in parentheses must produce a `bool` value
- No semicolon after parentheses
- Statement block executed/skipped based on condition
- Curly braces can be omitted if "statement block" is just **one** statement

```
if(<condition>)
{
  <statements>
}
```

```
if(<condition>)
  <statement>
```

```
if(age >= 18)
  Console.WriteLine("You can vote!");
Console.WriteLine("Goodbye");
```

If-Else Statements

- Choose *which* code to execute, depending on condition

```
if(age >= 18)
{
  Console.WriteLine("You can vote!");
}
else
{
  Console.WriteLine("You are too young to vote");
}
Console.WriteLine("Goodbye");
```

Condition

Code block to execute if condition is **true**

Note: no semicolon

Code block to execute if condition is **false**

After executing one of the two, continue here

If vs. If-Else

- if statement
  - If condition is true, code block is executed
  - If false, code block is skipped – nothing happens
- if-else statement
  - If condition is true, *first* block ("if block") is executed
  - If false, *second* block ("else block") is executed

```
if(<condition>)
{
  <statements>
}
```

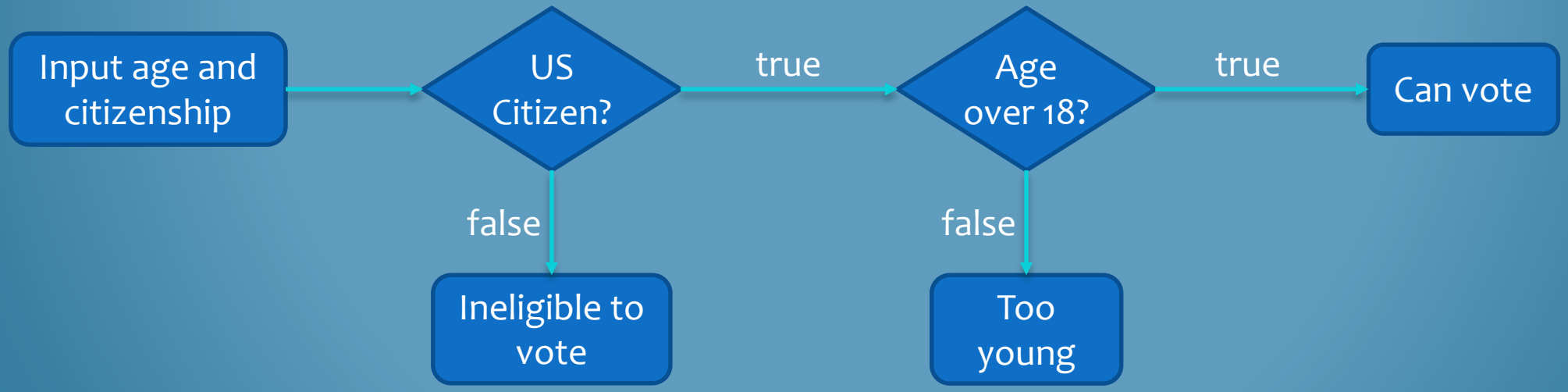
```
if(<condition>)
{
  <statements>
}
else
{
  <statements>
}
```

Outline

- If and if-else statements
- **Nested if statements**
- If-else-if statements

More Complex Decisions

- Once you know a condition is true/false, there might be a further condition to test
- Example with voting:



Nesting If Statements

- Code blocks can contain any statements, including if statements
- Within an if block, you know its condition is true

```
if(usCitizen == true)
{
  //At this point we know the user is a citizen
  //Now we need to determine if they are old enough
}
else
{
  Console.WriteLine("Sorry, only citizens can vote");
}
```

Nesting If Statements

```
if(usCitizen == true)
{
  if(age >= 18)
  {
    Console.WriteLine("You can vote!");
  }
  else
  {
    Console.WriteLine("You are too young to vote");
  }
}
else
{
  Console.WriteLine("Sorry, only citizens can vote");
}
```

Note: Can simply write `if(usCitizen)` since `usCitizen` is a `bool`

Either way, `usCitizen` is still true

Outline

- If and if-else statements
- Nested if statements
- **If-else-if statements**

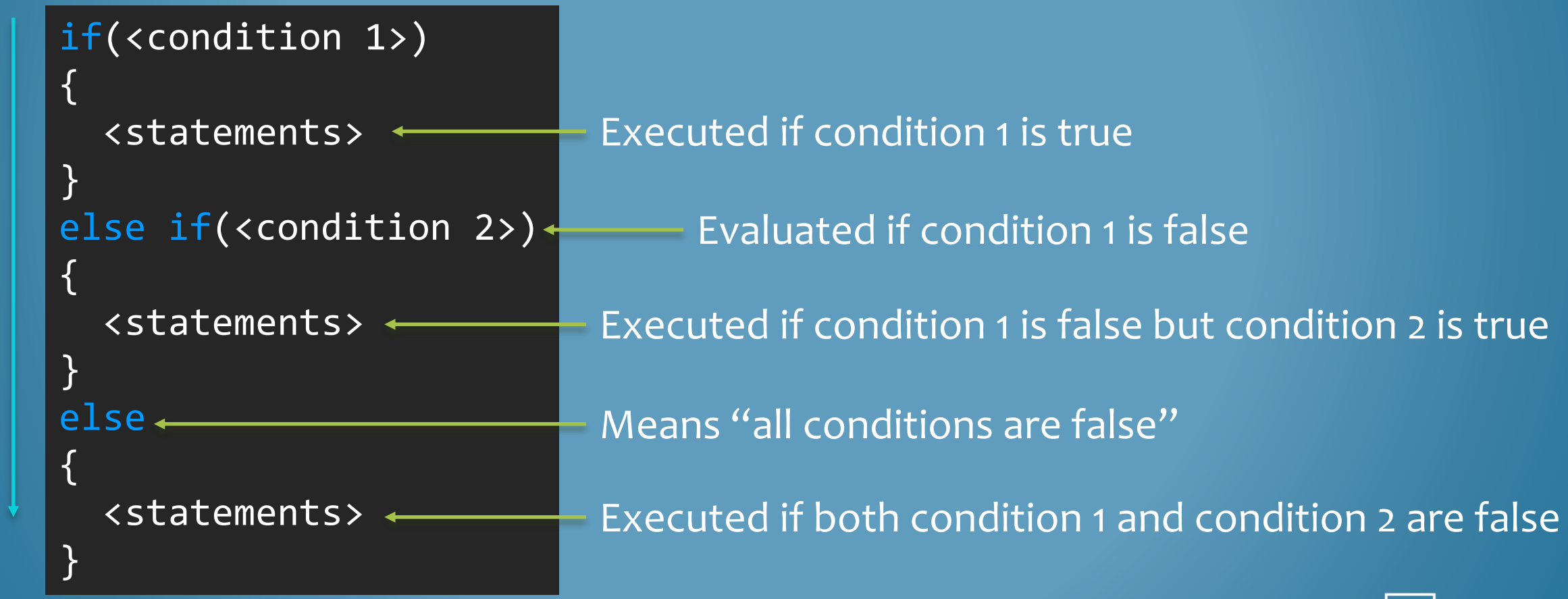
Mutually Exclusive Conditions

- Determine which floor a ClassRoom object is on
- If number is between 100 and 200, it's on the first floor
- If number is between 200 and 300, it's on the second floor
- If number is 300 or greater, it's on the third floor

```
class ClassRoom
{
  private string building;
  private int number;
...
  public int GetNumber()
  {
    return number;
  }
...
}
```

If-Else-If Syntax

- `else if`'s condition is evaluated if first `if`'s condition is false



Floors Problem

No need to write
`&& myRoom.GetNumber() < 300`
We know the first condition was false if this statement is being executed

No need for
`&& myRoom.GetNumber() < 200`

All conditions were false, so room number must be < 100

```
if(myRoom.GetNumber() >= 300)
{
  Console.WriteLine("Third floor");
}
else if(myRoom.GetNumber() >= 200)
{
  Console.WriteLine("Second floor");
}
else if(myRoom.GetNumber() >= 100)
{
  Console.WriteLine("First floor");
}
else
{
  Console.WriteLine("Basement?");
}
```

Conditions Can Be Different

- Else-if conditions don't need to use the same variable

```
int x;
if(myIntVar > 12)
  x = 10;
else if(myStringVar == "Yes")
  x = 20;
else if(myBoolVar)
  x = 30;
else
  x = 40;
```

(curly braces omitted to fit on the slide)

What value will x have in these situations?

myIntVar	myStringVar	myBoolVar	x
12	"Yes"	true	
15	"Yes"	false	
-15	"yes"	true	

If-Else-If vs. Nested If

- Some nested `if` statements are equivalent to `if-else-if`:

```
if(usCitizen == true)
{
  if(age >= 18)
  {
    Console.WriteLine("You can vote!");
  }
  else
  {
    Console.WriteLine("You are too young"
      + " to vote");
  }
}
else
{
  Console.WriteLine("Sorry, only citizens"
    + " can vote");
}
```

```
if(usCitizen == true && age >= 18)
{
  Console.WriteLine("You can vote!");
}
else if(usCitizen == true && age < 18)
{
  Console.WriteLine("You are too young"
    + " to vote");
}
else
{
  Console.WriteLine("Sorry, only citizens"
    + " can vote");
}
```

AUGUSTA UNIVERSITY

If-Else-If vs. Nested If

- Nested `if` statements with other code are harder to rewrite:

```
if(usCitizen)
{
  Console.WriteLine("How old are you?");
  int age = int.Parse(Console.ReadLine());
  if(age >= 18)
  {
    Console.WriteLine("You can vote!");
  }
  else
  {
    Console.WriteLine("You are too young to vote");
  }
}
else
{
  Console.WriteLine("Sorry, only citizens can vote");
}
```

age only exists within this block

Only ask for the user's age if they are a citizen

If-Else-If vs. Nested If

- An `if-else-if` statement can always be written as a nested `if`:
- `If-else-if` is easier to read, so prefer it when possible

```
if(myRoom.GetNumber() >= 300)
{
  Console.WriteLine("Third floor");
}
else
{
  if(myRoom.GetNumber() >= 200)
  {
    Console.WriteLine("Second floor");
  }
  else
  {
    if(myRoom.GetNumber() >= 100)
    {
      Console.WriteLine("First floor");
    }
  }
}
```

Summary

- If and if-else statements
- Nested if statements
- If-else-if statements