Revised: 12/22/2020

Logic and Branching

Michael C. Hackett

Assistant Professor, Computer Science

Community College of Philadelphia

Lecture Topics

- Boolean Logic and Expressions
  - Relational Operators and Expressions
  - Logical Operators and Expressions
  - Operator Precedence

- Branching
  - If Statements
  - Else Clauses
  - Elif Clauses
  - Nested If Statements
  - Inline Ifs

Relational Operators

- ***Relational operators*** perform a comparison that determines how two values relate to each other.
  - Each operator returns True or False

==	Equality Operator
!=	Inequality Operator
>	Greater Than Operator
<	Less Than Operator
>=	Greater Than or Equal To Operator
<=	Less Than or Equal To Operator

Relational Expressions

- A ***relational expression*** is an expression using a relational operator.
  - 1 == 5
  - 7 != 3
  - 16 > 5
  - 56 < 22
  - 10 >= 10
  - 9 <= 5
- A relational expression is a type of ***Boolean expression***.
  - A Boolean expression is one that evaluates to True or False.

Equality Operator ==

- Returns **true** if the operands are the same value.
- Returns **false** if the operands are different values.

```
i = 8
j = 10
result1 = i == j
```

false

```
k = 10
m = 10
result2 = k == m
```

true

Inequality Operator !=

- Returns **true** if the operands are different values.
- Returns **false** if the operands are the same value.

```
i = 8
j = 10
result1 = i != j
```

true

```
k = 10
m = 10
result2 = k != m
```

false

Greater Than Operator >

- Returns **true** if the first operand is larger than the second operand.
- Returns **false** if the first operand is equal to or smaller than the second operand.

```
i = 8
j = 10
k = 10
m = 11
result1 = i > j   false
result2 = j > k   false
result3 = m > i   true
```

Less Than Operator <

- Returns **true** if the first operand is smaller than the second operand.
- Returns **false** if the first operand is equal to or larger than the second operand.

```
i = 8
j = 10
k = 10
m = 11
result1 = i < j   true
result2 = j < k   false
result3 = m < i   false
```

Greater Than or Equal To Operator >=

- Returns **true** if the first operand is equal to or larger than the second operand.
- Returns **false** if the first operand is smaller than the second operand.

```
i = 8
j = 10
k = 10
m = 11
result1 = i >= j   false
result2 = j >= k   true
result3 = m >= i   true
```

Less Than or Equal To Operator <=

- Returns **true** if the first operand is equal to or smaller than the second operand.
- Returns **false** if the first operand is larger than the second operand.

```
i = 8
j = 10
k = 10
m = 11
result1 = i <= j   true
result2 = j <= k   true
result3 = m <= i   false
```

Logical Operators

- A ***logical operator*** connects two or more Boolean expressions or values into one **True** or **False** result.
  - Or, in the case of the logical not operator, reverse the logic of a Boolean expression or value.

`and` **`or`** **`not`**

- All three operators are keywords.
- A ***logical expression*** is an expression using a logical operator.

AND

- Evaluates to true if and only if both Boolean expressions are true.
- AND Truth Table:

B_1	B_2	B_1 and B_2
FALSE	FALSE	FALSE
FALSE	TRUE	FALSE
TRUE	FALSE	FALSE
TRUE	TRUE	TRUE

And Operator

```
b1 = False
b2 = False
```

false

```
result = b1 and b2
```

B_1	B_2	B_1 and B_2
FALSE	FALSE	FALSE
FALSE	TRUE	FALSE
TRUE	FALSE	FALSE
TRUE	TRUE	TRUE

And Operator

```
b1 = False
b2 = True
```

false

```
result = b1 and b2
```

B_1	B_2	B_1 and B_2
FALSE	FALSE	FALSE
FALSE	TRUE	FALSE
TRUE	FALSE	FALSE
TRUE	TRUE	TRUE

And Operator

```
b1 = True
b2 = False
```

false

```
result = b1 and b2
```

B_1	B_2	B_1 and B_2
FALSE	FALSE	FALSE
FALSE	TRUE	FALSE
TRUE	FALSE	FALSE
TRUE	TRUE	TRUE

And Operator

```
b1 = True
b2 = True

                      true
result = b1 and b2
```

B_1	B_2	B_1 and B_2
FALSE	FALSE	FALSE
FALSE	TRUE	FALSE
TRUE	FALSE	FALSE
TRUE	TRUE	TRUE

OR

- Evaluates to true if at least one of the Boolean expressions is true.
- OR Truth Table:

B_1	B_2	B_1 or B_2
FALSE	FALSE	FALSE
FALSE	TRUE	TRUE
TRUE	FALSE	TRUE
TRUE	TRUE	TRUE

Or Operator

```
b1 = False
b2 = False

                    false
result = b1 or b2
```

B_1	B_2	B_1 or B_2
FALSE	FALSE	FALSE
FALSE	TRUE	TRUE
TRUE	FALSE	TRUE
TRUE	TRUE	TRUE

Or Operator

```
b1 = False
b2 = True

                    true
result = b1 or b2
```

B_1	B_2	B_1 or B_2
FALSE	FALSE	FALSE
FALSE	TRUE	TRUE
TRUE	FALSE	TRUE
TRUE	TRUE	TRUE

Or Operator

```
b1 = True
b2 = False

                    true
result = b1 or b2
```

B_1	B_2	B_1 or B_2
FALSE	FALSE	FALSE
FALSE	TRUE	TRUE
TRUE	FALSE	TRUE
TRUE	TRUE	TRUE

Or Operator

```
b1 = True
b2 = True

                    true
result = b1 or b2
```

B_1	B_2	B_1 or B_2
FALSE	FALSE	FALSE
FALSE	TRUE	TRUE
TRUE	FALSE	TRUE
TRUE	TRUE	TRUE

NOT

- Inverts/Negates a Boolean expression.
- NOT Truth Table:

B_1	not B_1
FALSE	TRUE
TRUE	FALSE

Not Operator

```
b1 = True

                    false
result = not b1
```

B_1	not B_1
FALSE	TRUE
TRUE	FALSE

Not Operator

```
b1 = False

                  true
result = not b1
```

B_1	not B_1
FALSE	TRUE
TRUE	FALSE

Logical Operator Precedence

1. **not** Operator
2. **and** Operator
3. **or** Operator

Logical Operator Precedence

```
b1 = False
b2 = True
b3 = False


result = not b1 or b2 and b3
```

- What is the value of the result variable?

Logical Operator Precedence

```
b1 = False
b2 = True
b3 = False
```

```
                      false      true       false
result = not b1 or b2 and b3

         True or b2 and b3

         True or False

            True
```

Operator Precedence

0.	**( )**	Expressions in parentheses are always evaluated first.
1.	**not**, **-**	Not Operator, Unary Negation (-5)
2.	******	Exponent
2.	*****, **/**, **//**, **%**	Multiplication, Float Division, Integer Division, Modulus
3.	**+**, **-**	Addition, Subtraction
4.	**<**, **>**, **<=**, **>=**	Less than (or equal), Greater than (or equal)
5.	**==**, **!=**	Equal to, Not equal to
6.	**and**	And Operator
7.	**or**	Or Operator
8.	**=**, **+=**, **-=**, ***=**, **/=**, **%=**	Assignment and Combined Assignment

Operator Precedence

```
num1 = 4
num2 = 5
b1 = False

result = not b1 and num1 + num2 >= 9
```

- What is the value of the result variable?

Operator Precedence

```
num1 = 4
num2 = 5
b1 = False

result = not b1 and num1 + num2 >= 9
         True and num1 + num2 >= 9
         True and 9 >= 9
         True and True
              True
```

Operator Precedence

```
num1 = 4
num2 = 5
b1 = False

result = b1 or num1 + num2 == 9
```

- What is the value of the result variable?

Operator Precedence

```
b1 = False
b2 = False
b3 = False

result = not b1 or b2 and b3
```

- What is the value of the result variable?

Operator Precedence

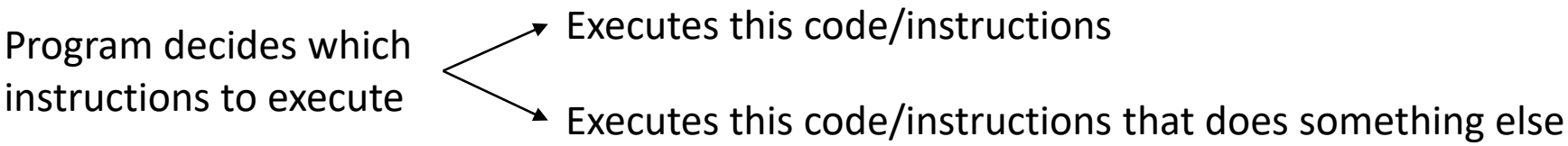
```
b1 = False
b2 = False
b3 = False

result = not(b1 or b2) and b3
```

- What is the value of the result variable?

Branching

- ***Branching***, in computer science, is when a computer program or algorithm departs from executing its current set of instructions to begin executing different instructions.

- In programming terms, branching normally refers to when a program or algorithm *decides* which set of instructions to execute.



If Statements

- An ***if statement*** tests a Boolean expression and will only execute its instructions if the expression evaluates to true.
  - The code will be "skipped" if the Boolean expression evaluates to false.
- The syntax for an if statement in Python is shown below.

```
if Boolean Expression :
  #code that will be
  #executed if the Boolean Expression
  #evaluates to True
```

Indent one tab.

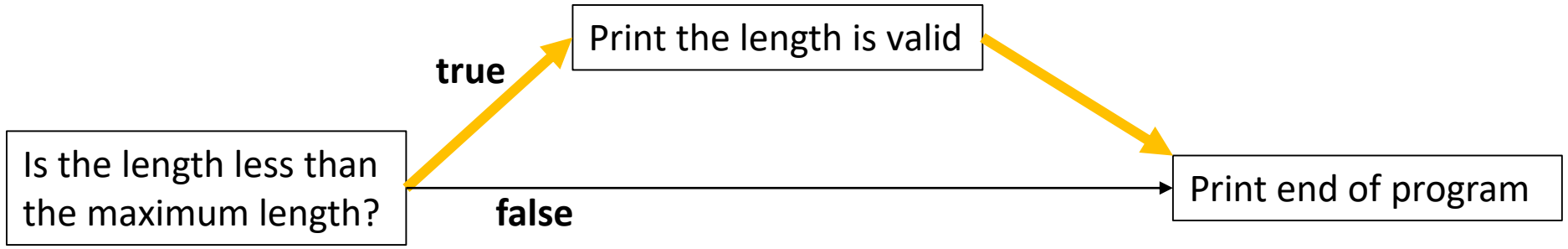
- The Boolean expression as part of an if statement forms a ***conditional expression***.

If Statements

```
length = 80
max_length = 100

if length < max_length :
  print("This is a")
  print("valid length.")

print("End of program.")
```

true

```
This is a
valid length.
End of program.
```

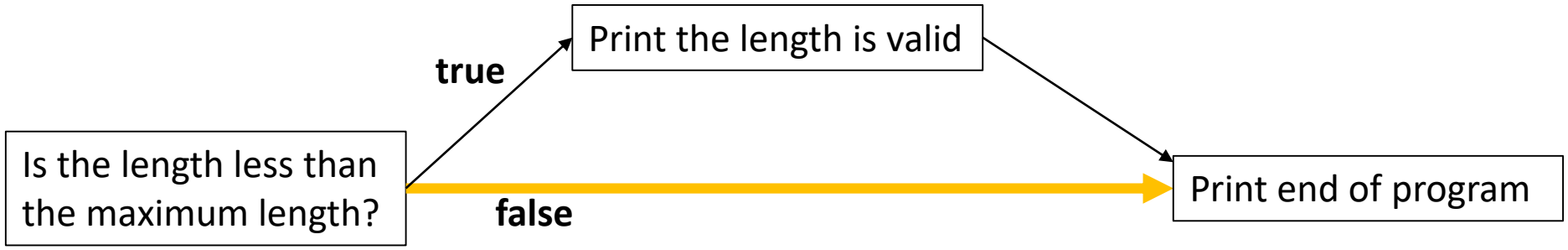


If Statements

```
length = 180
max_length = 100

if length < max_length :
  print("This is a")
  print("valid length.")

print("End of program.")
```

false

```
End of program.
```



If Statements

```
length = 50
max_length = 100

if length >= 0 and length < max_length :
  print("This is a")
  print("valid length.")

print("End of program.")
```

```
This is a
valid length.
End of program.
```

If Statements

```
if b1 == True :
```

Is equivalent to

```
if b1 :
```

```
if b1 != True :
```

Is equivalent to

```
if not b1 :
```

Else Clauses

- An ***else clause*** is a set of instructions that will only execute when its associated if statement's Boolean expression evaluates to false.
- The syntax for an else clause in Python is shown below.

```
if Boolean Expression :
   #code that will be
   #executed if the condition
   #evaluates to True
else :
   #code that will be
   #executed if the condition
   #evaluated to False
```

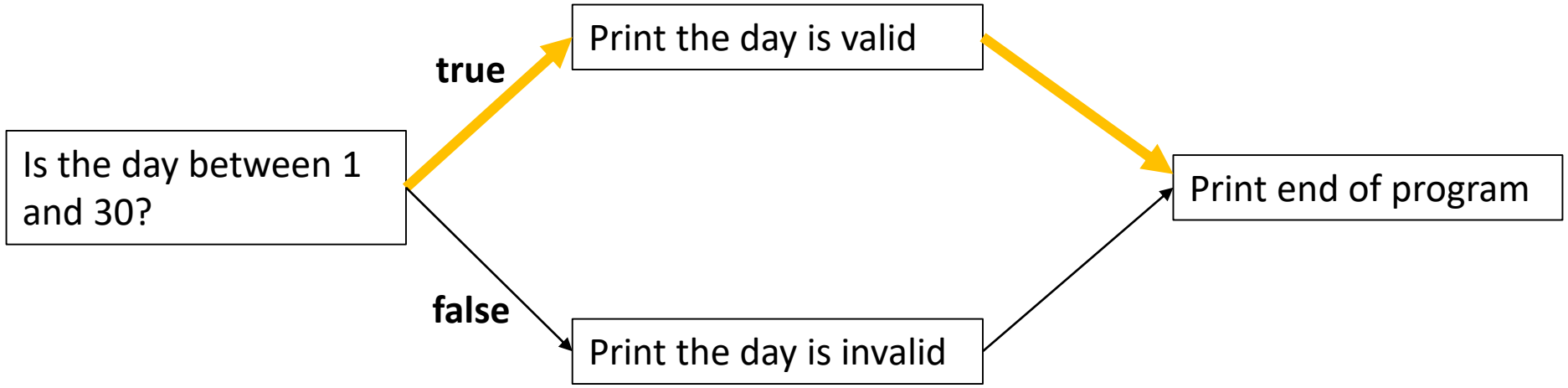
Else Clauses

```
    day = 10

true if day > 0 and day <= 30 :
       print("This is a valid")
       print("day in September.")
    else :
       print("This is not a valid")
       print("day in September.")

    print("End of program.")
```

```
This is a valid
day in September.
End of program.
```

Else Clauses

`day = 10`



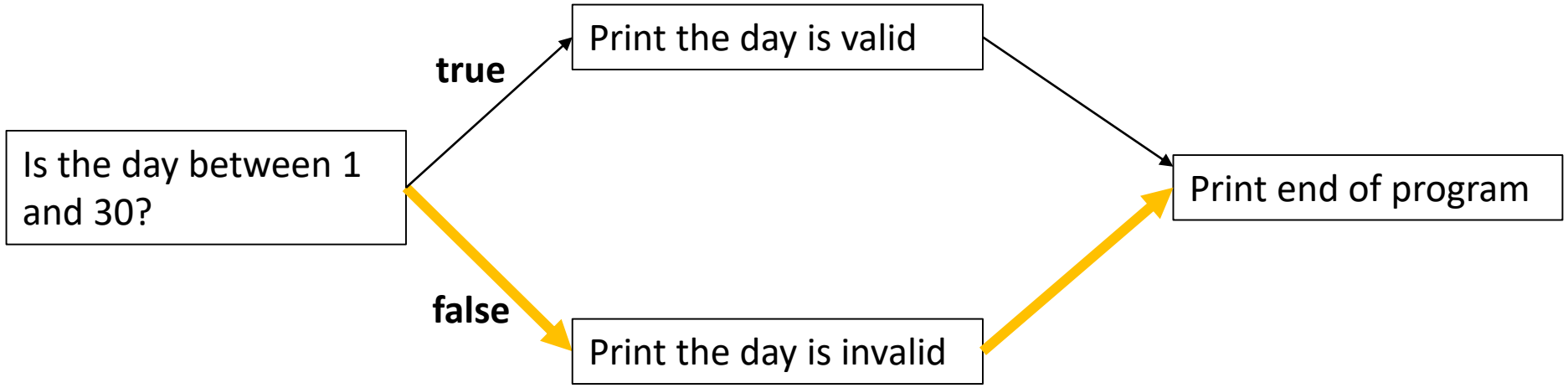
Else Clauses

```
    day = 31

false if day > 0 and day <= 30 :
        print("This is a valid")
        print("day in September.")
      else :
        print("This is not a valid")
        print("day in September.")

      print("End of program.")
```

```
This is not a valid
day in September.
End of program.
```

Else Clauses

day = 31



Elif Clauses

- An ***elif clause*** (short for “else if”) is an additional if statement that allows testing alternative Boolean expressions.
- The syntax for an elif clause in Python is shown below.

```
if Boolean Expression 1 :
   #code that will be executed if the expression
   #evaluates to True
elif Boolean Expression 2 :
   #code that will be executed if Boolean Expression 1 was False
   #and this Boolean Expression 2 evaluates to True
else :
   #code that will be executed if no previous expressions
   #evaluated to True
```

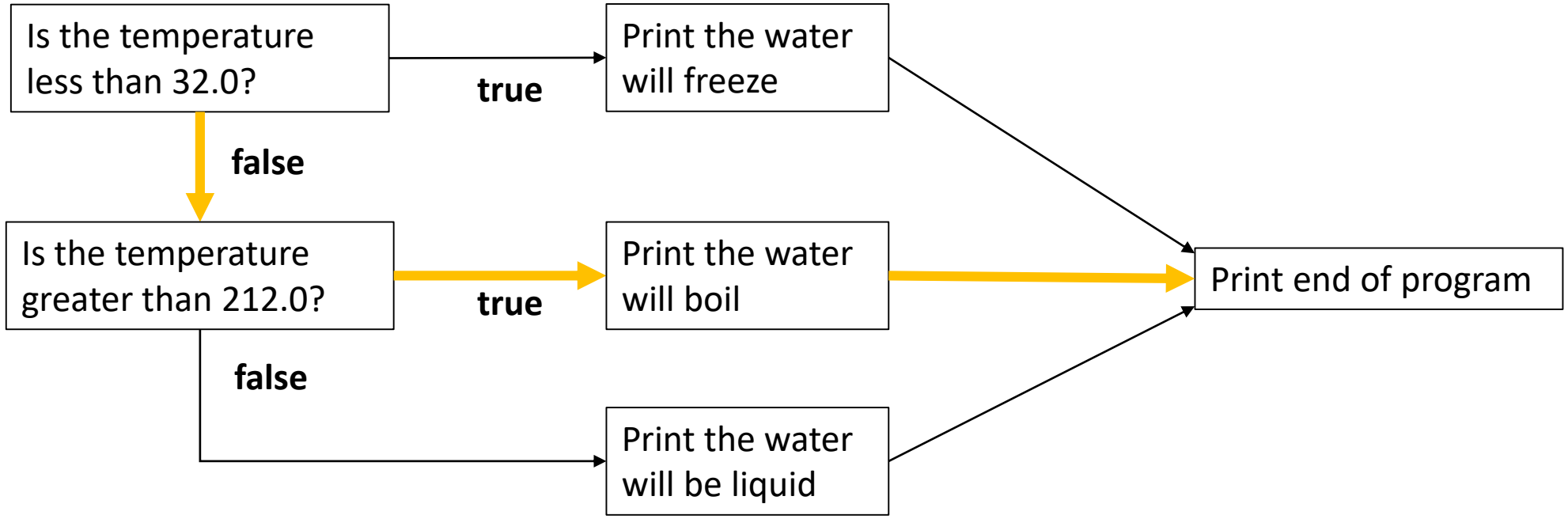
Elif Clauses

```
    temp = 215.5

false if temp <= 32.0 :
        print("Water will freeze.")
true  elif temp >= 212.0 :
        print("Water will boil.")
      else :
        print("Water will be liquid.")

      print("End of program.")
```

```
Water will boil.
End of program.
```

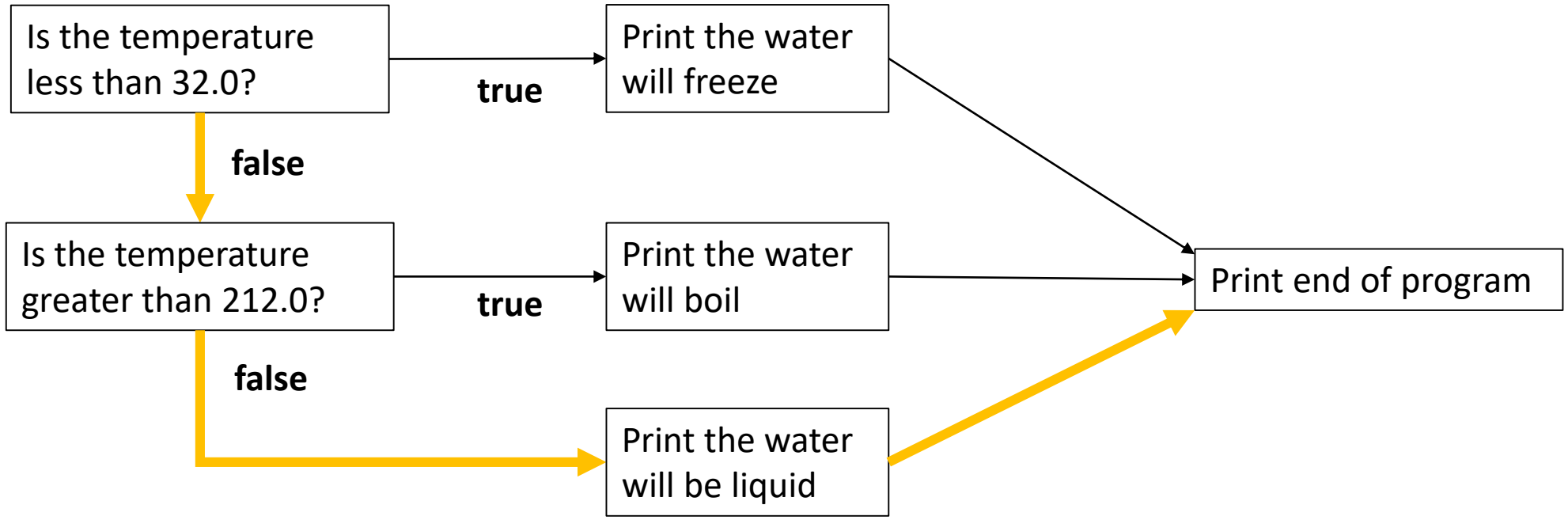
Elif Clauses

`temp = 215.5`



Elif Clauses

`temp = 55.7`



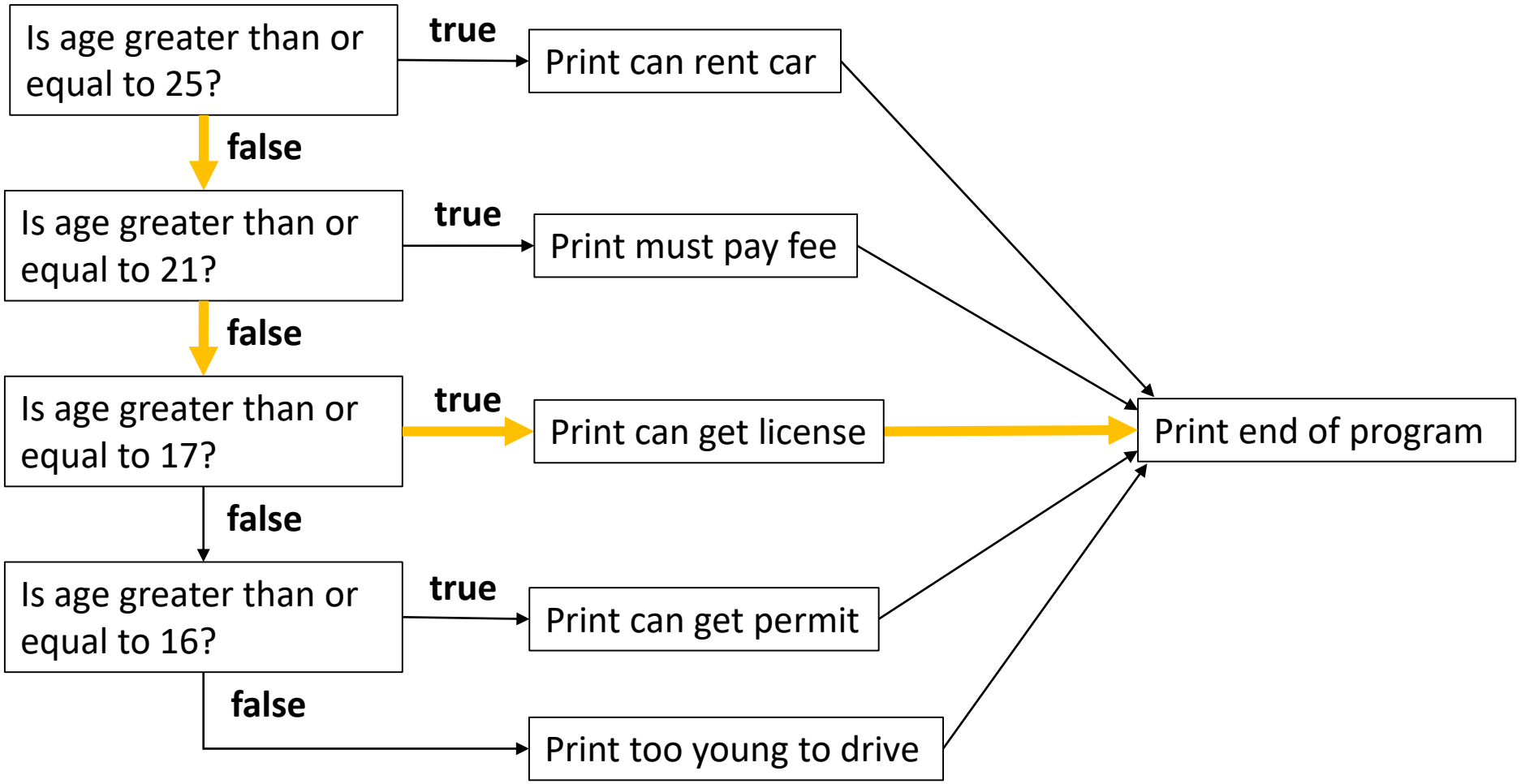
Elif Clauses

```
    age = 19

false if age >= 25 :
        print("Can rent a car.")
false elif age >= 21 :
        print("Must pay underage driver fee.")
true  elif age >= 17 :
        print("Can get a license.")
      elif age >= 16 :
        print("Can get a permit.")
      else :
        print("Too young to drive.")

      print("End of program.")
```

```
Can get a license.
End of program.
```

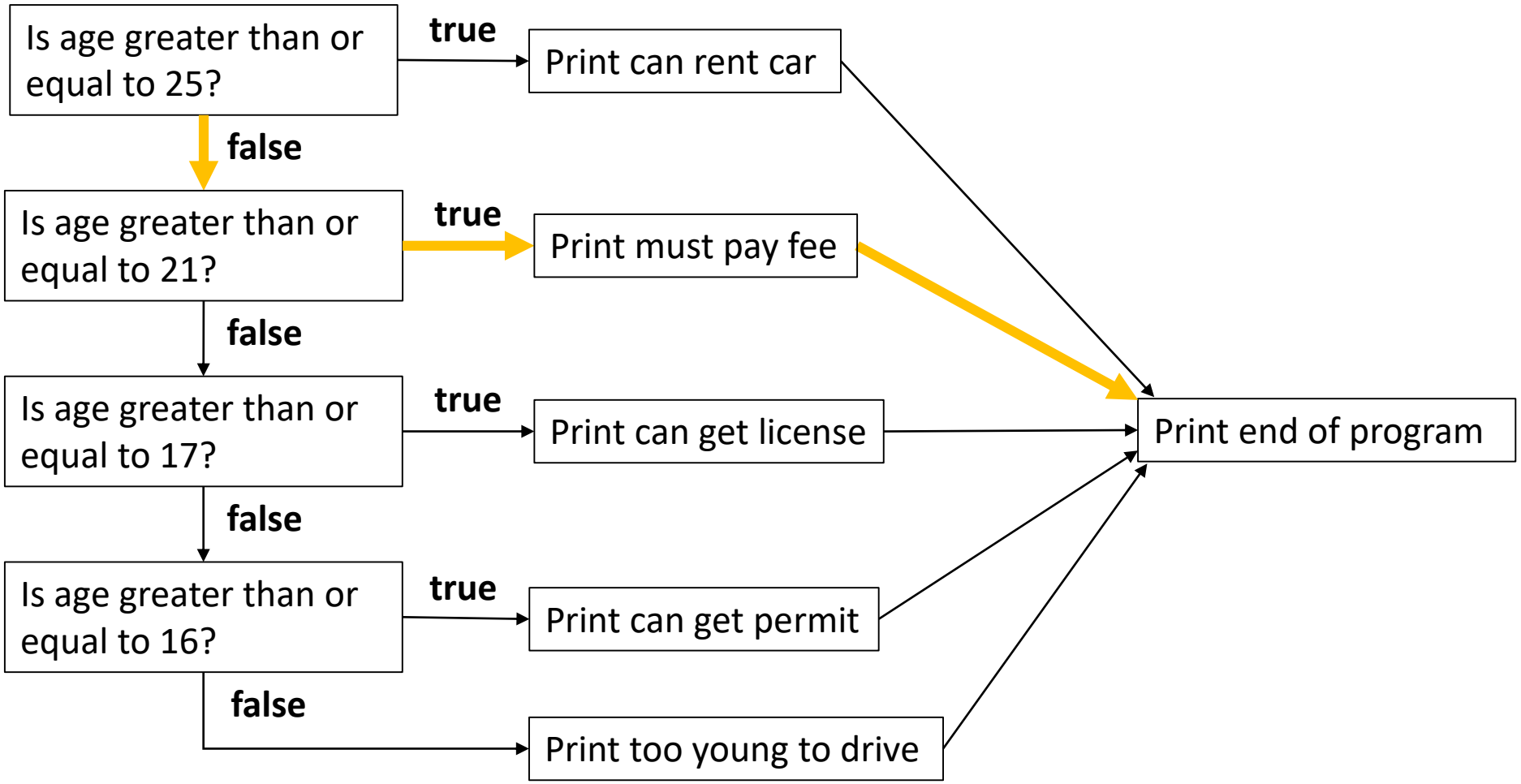
Elif Clauses

age = 19



Elif Clauses

age = 23



If Statement and Elif/Else Clause Rules

- If Statements
  - **Must** always be first.
  - May be followed by any number of elif statements.
  - May be followed by one else statement.
- Elif Clauses
  - Optional.
  - **Must** follow an if statement or elif clause.
  - No limit to the number of elif clauses.
  - May be followed by one else clauses.
- Else Clauses
  - Optional.
  - **Must** follow an if statement or elif clause.
  - Only one else clause.
  - **Always** the last clause.

Nested If Statements

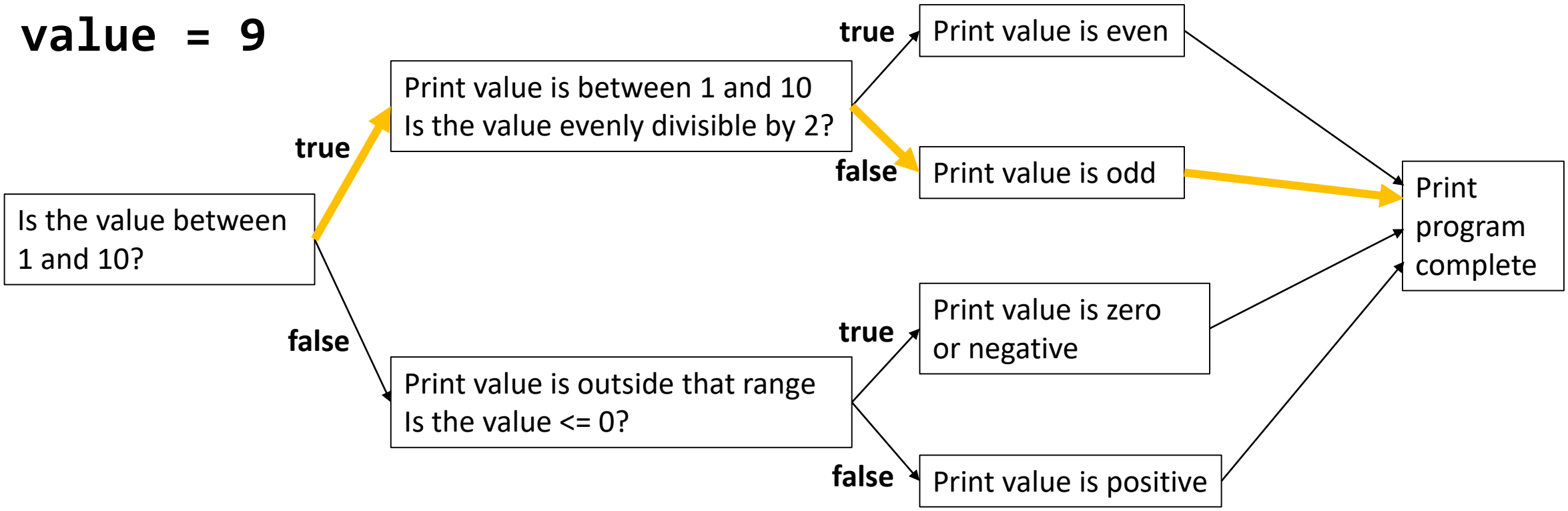
- A ***nested if statement*** is an if statement within the body of an if statement or else clause.
- Be sure you indent properly.

```
if Boolean Expression 1 :
    #code that will be executed if expression 1 is true
    if Boolean Expression 2 :
        #code that will be executed if expression 2 is true
    else :
        #code that will be executed if expression 2 is false
else :
    #code that will be executed if expression 1 is false
```

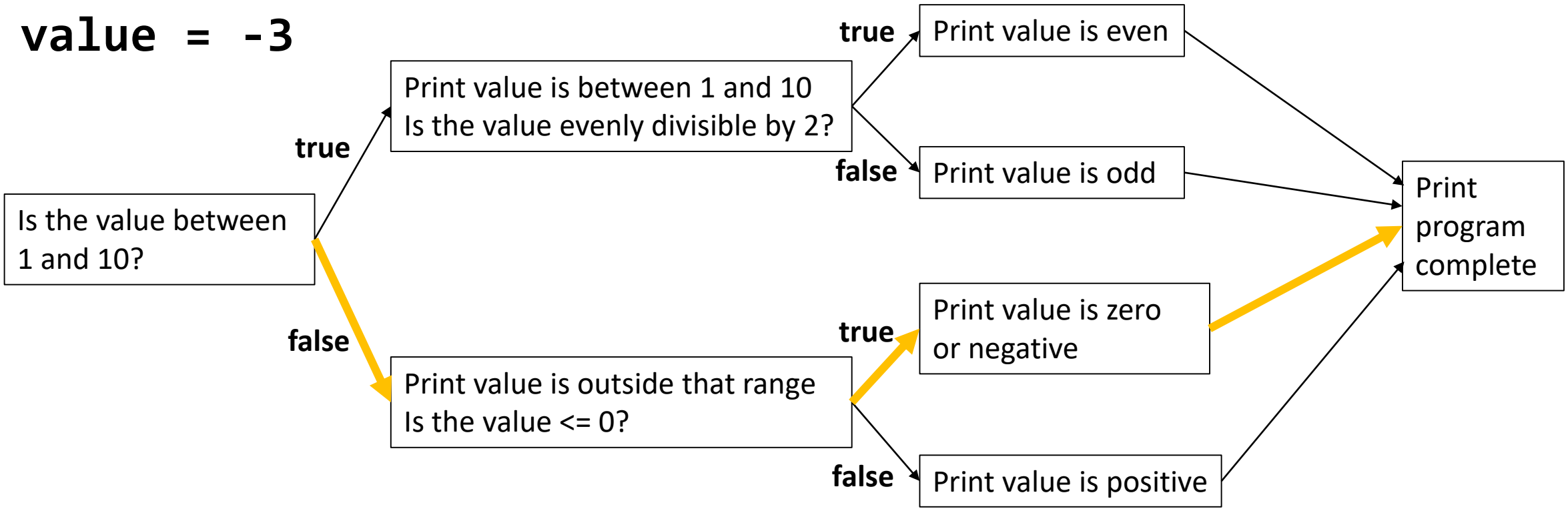
Nested If Statements

```
value = 9

if value >= 1 and value <= 10 :
  print("Your value is between 1 and 10.")
  if value % 2 == 0 :
    print("Your value is even.")
  else :
    print("Your value is odd.")
else :
  print("Your value is outside the range of 1 and 10.")
  if value <= 0 :
    print("Your value is zero or negative.")
  else :
    print("Your value is more than 10.")

print("Program complete")
```

Nested If Statements



Nested If Statements



Inline If Statements

- Shorthand if-else statement.
- Uses three operands.
  - Sometime called a ternary operation

- Syntax:

do this `if` ***condition*** `else` ***do this instead***

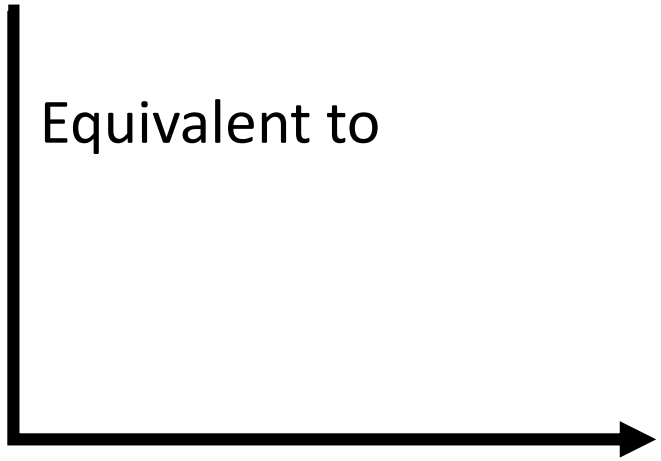
Inline If Statements

```
number1 = int(input("Enter the first number: "))
number2 = int(input("Enter the second number: "))

largest = number1 if number1 > number2 else number2
print("The larger number is", largest)
```

```
Enter the first number: 7
Enter the second number: 3
The larger number is 7
```

Inline If Statements

```
largest = number1 if number1 > number2 else number2
```




```
if number1 > number2 :
    largest = number1
else :
    largest = number2
```

Inline If Statements

```
number1 = int(input("Enter the first number: "))
number2 = int(input("Enter the second number: "))

largest = number1 if number1 > number2 else number2
print("The larger number is", largest)
```

```
Enter the first number: 4
Enter the second number: 9
The larger number is 9
```

Inline If Statements

```
number1 = int(input("Enter the first number: "))
number2 = int(input("Enter the second number: "))

largest = str(number1) if number1 > number2 else str(number2)
print("The larger number is" + largest)
```

```
Enter the first number: 6
Enter the second number: 18
The larger number is 18
```

Inline If Statements

```
number1 = int(input("Enter the first number: "))
number2 = int(input("Enter the second number: "))

largest = str(number1 if number1 > number2 else number2)
print("The larger number is" + largest)
```

```
Enter the first number: 14
Enter the second number: 12
The larger number is 14
```