CS65: Introduction to Computer Science

Conditional Statements
Boolean Expressions

Md Alimoor Reza
Assistant Professor of Computer Science

Announcements

Lab #1 due today (yesterday)

Lab #2 out today; due on Tuesday, September 23th

Text Reading

https://www.brianheinold.net/python/python_book.html#chapter_ifstatements

Review from last lecture

- Expressions
- Data types
- Receiving input from user
  - Counterpart of showing an user some outputs

Recap: Assignment Statements

Code with a = is called an *assignment statement*

You use an *assignment statement* to create a variable and make it reference a piece of data

Examples:

```
age = 25
hourly_rate = 15.5
greeting = "Hello everyone"
```

Whatever is on the right side of = will be evaluated and saved to the variable named on the left side.

Recap: Types

Types

All data has a ***type*** which defines what kind of thing it is and what you can do with it.

Types we've seen so far:

```
3
4 num_students = 42 # whole numbers
5 avg_gas_estimate = 2.70 # numbers with decimal
6 print("Hello world!") # text data (strings)
```

Recap: Types

What do you think will happen here?

types2.py

types2.py

```
1  # Alimoor Reza
2  # Introduction to data types
3
4  val = "32"
5  print( type(val) )
6  print( type( len(val) ) )
7
```

Recap: len() function

Other Operations

The `len` function will output the number of characters that a string has

```
>>> len("hello")
5
>>> len("123")
3
>>> len("oh my gosh this is a very long string")
37
```

Recap: int() and float() functions

Other Operations

The `int` function will convert the number/string into an integer number

other_functions.py

```
1  # alimoor reza
2  # int() function and float() function
3
4  int_num = int(4.5)
5  print(int_num)
```

4

The `float` function will convert a number/string into an decimal number

other_functions.py

```
1  # alimoor reza
2  # int() function and float() function
3
4  float_num = float(4)
5  print(float_num)
```

4.0

Recap: user input

User Input

Python provides a function called `input` that gets input from the keyboard.

When this function is called, the program *stops and waits* for the user to type something.

When the user presses Return or Enter, the program resumes and input returns what the user typed as a string

input_and_output.py

```
input_and_output.py ×
1  # author: Alimoor Reza
2  # description: receiving input from the user and displaying output
3
4  first_name = input("Please enter your first name")
5  last_name = input("Please enter your last name")
6  print("Hello ", first_name, " ", last_name)
```

Recap: user input

Exercise #1:

$$C = (F - 32) * \frac{5}{9}$$

Create a file called temperature.py

Type in the following code

temparature.py

```
1  # Alimoor Reza
2  # Introduction to data types
3
4  # formula: C = (F-32)*(5/9)
5
6  fahrenheit_temp = input("Enter the temparature in fahrenheit scale:
7  celcius_temp = (fahrenheit_temp-32)*(5/9)
8  print("Converted temparature is: ", celcius_temp, " in Celcius")
```

Will it work? Why or why not??

Fix it.

Recap: user input

Converting a string to an integer

Fortunately, there's a built-in function we can use to convert something to an integer: **`int()`**

```
4  # formula: C = (F-32)*(5/9)
5
6  fahrenheit_temp_str = input("Enter the temparature in fahrenheit sca
7  fahrenheit_temp_int = int(fahrenheit_temp_str)
8  celcius_temp = (fahrenheit_temp_int-32)*(5/9)
9  print("Converted temparature is: ", celcius_temp, " in Celcius")
10
```

Today's content

- Division operators

- Conditional statements
  - if
  - if-else
  - if-elif-else
    - multiple selection statements

Division Operations

Floating-point division operator is simply this: /

The / operator will always give you a float as a result.

Division Operations

Floating-point division operator is simply this: /

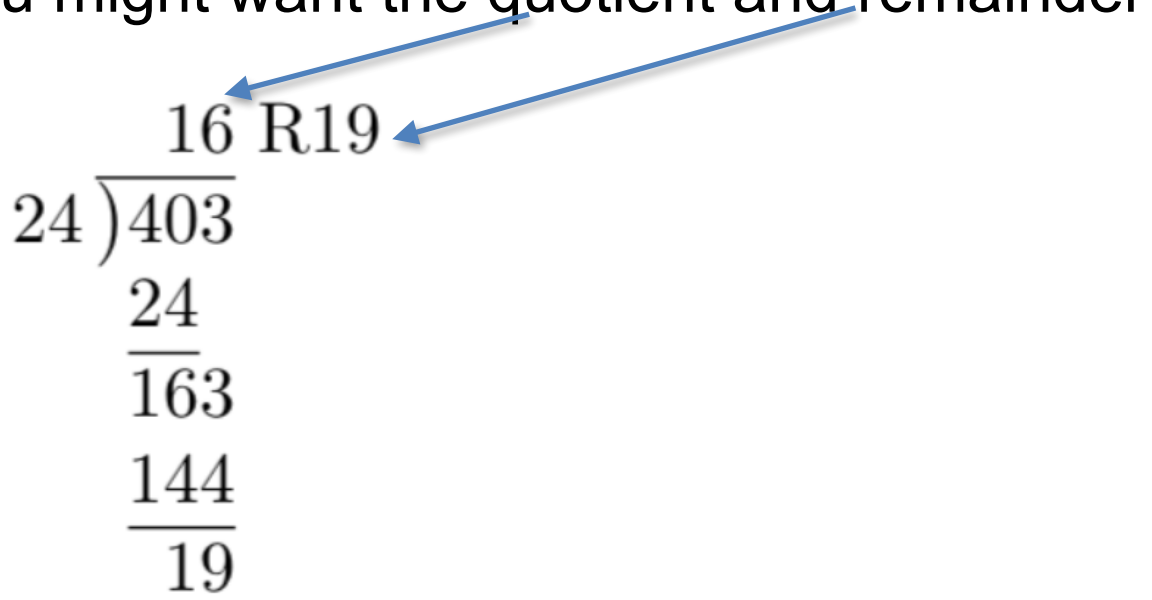
The / operator will always give you a float as a result.

```
>>> 320/20
16.0
>>> 403.0/24.0
16.791666666666668
```

```
>>> val = 4/2
>>> print(val)
 2.0
```

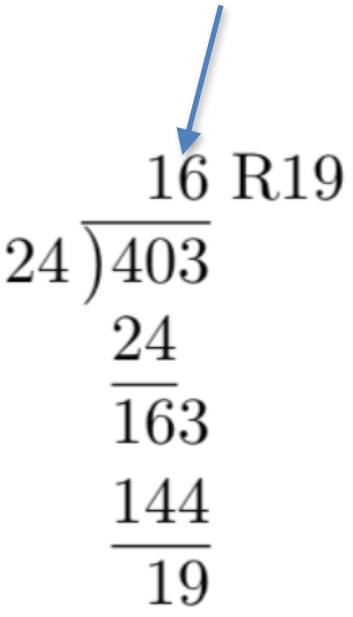
Integer Division Operations

Sometimes, instead of getting the full floating-point expansion, you might want the quotient and remainder separately.



Integer Division Operation //

The // is the floor division operator - it gives you the quotient and ignores and fractional/remainder part.




```
>>> 403//24
16
```

Integer Division Operator %

The % operator is the modulo operator - another word for remainder - it will ignore the quotient and just give you the remainder.

```
>>> 403%24
19
```

$$
\begin{array}{r}
16\ \mathrm{R}19 \\
24\overline{)403} \\
\underline{24} \\
163 \\
\underline{144} \\
19
\end{array}
$$

Try it!

Try some of the following in your interactive shell. What pattern do you notice?

```
0 // 2
1 // 2
2 // 2
3 // 2
4 // 2
5 // 2

20 // 5
21 // 5
22 // 5
23 // 5
24 // 5
25 // 5
```

```
0 % 2
1 % 2
2 % 2
3 % 2
4 % 2
5 % 2

20 % 5
21 % 5
22 % 5
23 % 5
24 % 5
25 % 5
```

Good quiz questions…

What is the difference between // and /?

What is the difference between // and %?

When is this useful?

Modulo (%) is also good if you need to check if a number is even or odd

```
>>> 253 % 2 #odd numbers have a remainder of 1 when divided by 2

1
```

```
>>> 254 % 2 #even numbers have a remainder of 0 when divided by 2

0
```

When is this useful?

- Or, if you want to check if a number is divisible by 4 - say you're checking to see if the year is a presidential election year, Olympic year, leap year, etc., you can see if the result is 0

```
>>> 2024 % 4
0
```

When is this useful?

- There are 100 pieces of candy; there are 8 people at the party; how many pieces of candy does each person get? How much candy is left over?
- I'm programming a super computer with 64 processors to perform 1500 variations of a simulation for a physics experiment, and each of the simulations takes about the same amount of time to run. How many simulations should be given to each processor?
- I have 78 cents, how many quarters should I give in change?

Show: Change.py

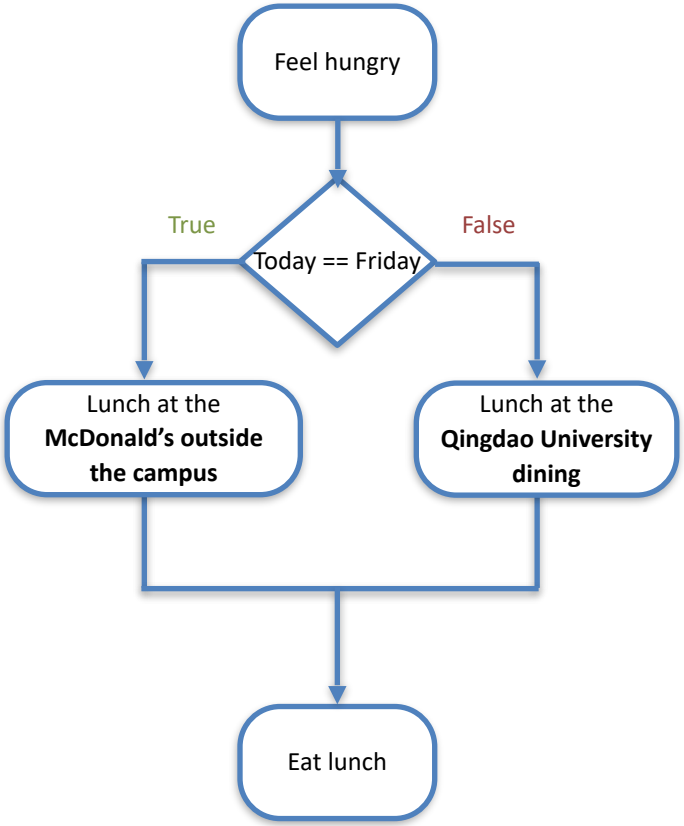
Roadmap

- Division operators

- Conditional statements
  - if
  - if-else
  - if-elif-else
    - multiple selection statements

Motivation

- program taking one *path* or *branch* of the code instead of taking another, based on the **boolean expression**'s value
- It allows us to ask true/false questions in our code. Depending on the boolean answer (True or False), the program will execute a specific branch.



If statements (conditionals)

Sometimes, you want your program to do something different based on different conditions. These are called conditional statements.

Also known as "*if statements*"

Motivating Example

For example, consider a pay calculator. If an employer offers 50% extra pay for overtime (i.e., worked > 40 hours), how could we account for that?

```python
# Alimoor Reza
# Date: 09/18/25
# if statement

salary_per_hour = 10
total_hours     = 45

'''
option 1: only correct if he worked
          less than 40 hours
'''
total_salary = salary_per_hour*total_hours

'''
option 2: only correct if he worked for
          more than 40 hours
'''
# extra_hours = total_hours - 40
# total_salary = salary_per_hour*40 + (salary_per_hour*1.5*extra_hours)

print("Total salary: ", total_salary)

```

```
>>> %cd /Users/reza/Class_and_Research/drake_university/c
       cs65_fall25/cs65_codes/lecture4
>>> %Run if_statement.py

  Total salary:  450

>>>
```

Syntax for if statements

- keyword `if`
- a condition, something that can be *true* or *false*
- colon `:`
- *indented* block of code
  - All of the code that is indented will be executed if the condition is true

Syntax for if statements

- keyword **`if`**
- a condition, something that can be *true* or *false*
- colon **:**
- *indented* block of code
  - All of the code that is indented will be executed if the condition is true

if_statement.py

if_statement.py

```
1  # Alimoor Reza
2  # 09/18/2025
3  # simple if statement
4
5
6  num = int(input("Enter a number please: "))
7
8  if num == 5:
9      print("You won the lottery ...")
10
11 print("Thank you!")
```

`if` Statement

if_statement.py

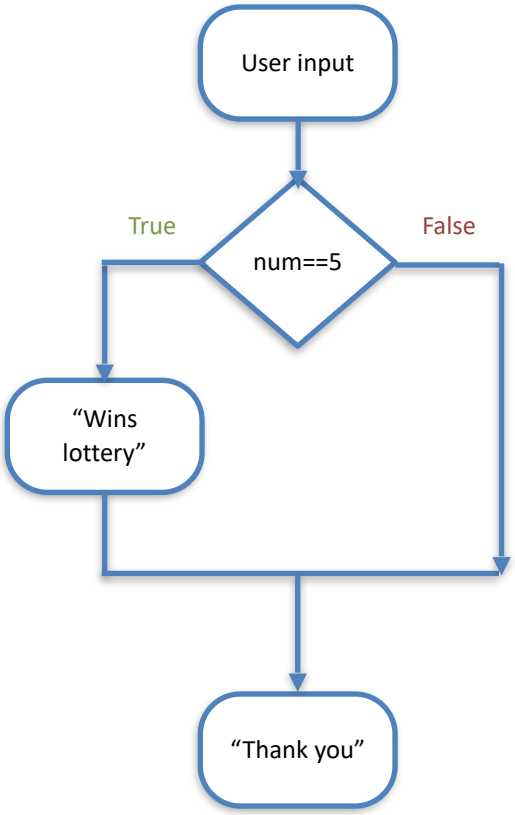
```
1  # Alimoor Reza
2  # 09/18/2025
3  # simple if statement
4
5
6  num = int(input("Please, enter a number. "))
7
8  if num == 5:
9      print("Yeah! I won a lottery ...")
10
11 print("Thank you!")
```

Shell

```
>>> %Run if_statement.py

  Please, enter a number. 5
  Yeah! I won a lottery ...
  Thank you!

>>>
```



If statements and If-else statements

An if-statement will only execute the indented code if the condition is True

If the condition is False, the code block is *not* executed.

IF you want to have something different happen when the condition is False, then you can add an `else` clause.

Introducing,.... The If-Else statement

`if ... else` Statement

if_else_statement1.py

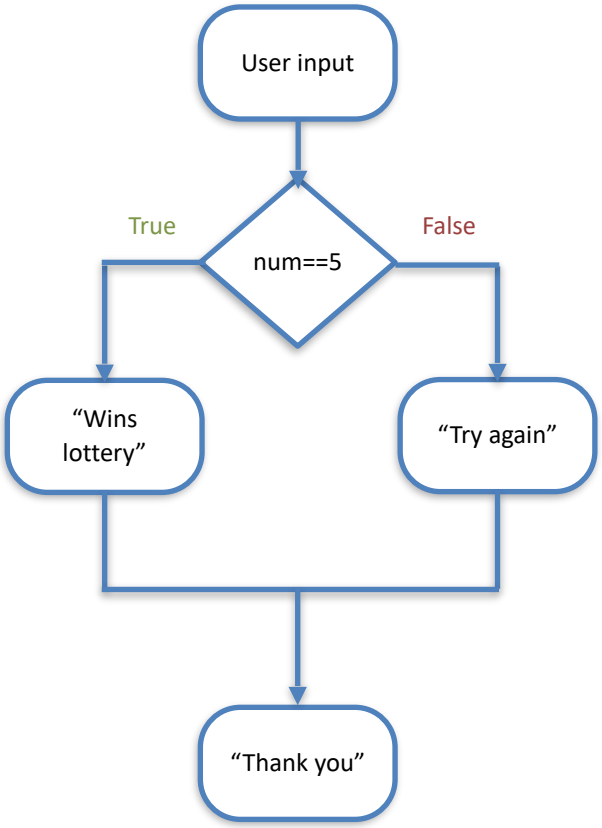
```
1  # Alimoor Reza
2  # 09/18/2025
3  # simple if-else statement
4
5
6  num = int(input("Please, enter a number. "))
7
8  if num == 5:
9      print("Yeah! I won a lottery ...")
10 else:
11     print("Oh gosh! better luck next time ...")
12
13 print("Thank you!")
14
```

Shell

```
>>> %Run if_else_statement1.py

  Please, enter a number. 6
  Oh gosh! better luck next time ...
  Thank you!

>>>
```



`if ... else` Statement

if_else_statement2.py

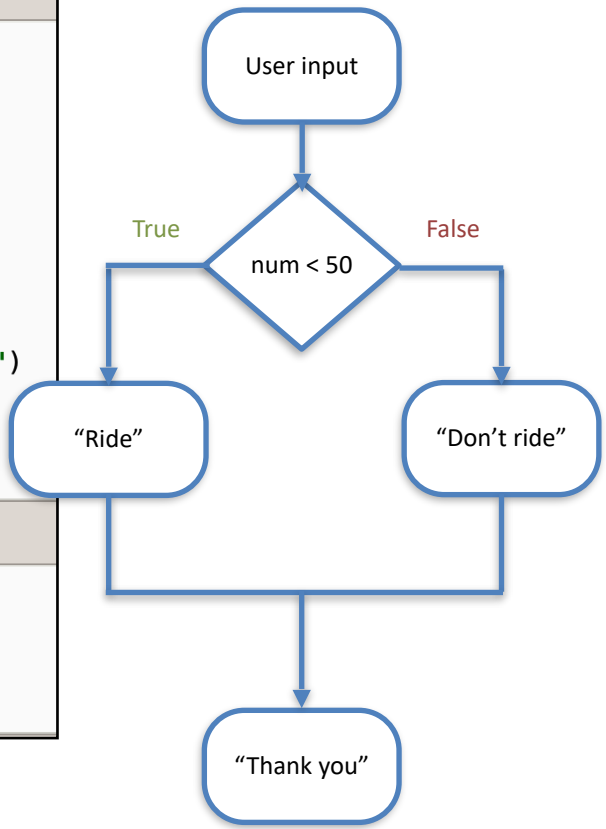
```
1  # Alimoor Reza
2  # 09/18/2025
3  # simple if-else statement
4
5
6  num = int(input("Please, enter your height in inches: "))
7
8  if num < 50:
9      print("You are not tall enough for this ride ...")
10 else:
11     print("You are not tall enough for this ride. Enjoy your ride ...")
12
13 print("Thank you!")
14
```

Shell

```
>>> %Run if_else_statement2.py

  Please, enter your height in inches: 50
  You are not tall enough for this ride. Enjoy your ride ...
  Thank you!

>>>
```



Syntax for if-else statements

- keyword **`if`**
- a condition, something that can be *true* or *false*
- colon `:`
- *indented* block of code
  - All of the code that is indented here will be executed if the condition is **True**
- keyword **`else`**
- colon `:`
- indented block of code
  - All of the code that is indented here will be executed if the condition is **False**

exercise1.py

Exercise #1

For example, consider a pay calculator. If an employer offers 50% extra pay for overtime (i.e., worked > 40 hours), how could we account for that?

How can we make this work using an if-else?

```
1  # your name here
2  # exercise... fix this with an if-else statement
3
4  wage = float(input("Enter your hourly wage: "))
5  hours = float(input("Enter your number of hours worked: "))
6
7  #option 1: only correct if they didn't work overtime
8  pay = wage*hours
9
10 #option 2: only correct if they worked overtime
11 #overtime_hours = hours-40
12 #pay = (wage*40) + (wage*1.5*overtime_hours)
13
14 print("Total pay: ",pay)
```

Boolean type

The Boolean type has two possible values: **`True`** and **`False`**

```
bool_variable = True

if bool_variable:
    print("The condition it True!")
```

Conditions and Booleans

The condition for an if/if-else statement can be anything that results in a Boolean.

An expression, like `height < 48` that results in a Boolean is called a *Boolean expression.*

Some Boolean operators

- < less than
- > greater than
- <= less than or equal
  - note there is no ≤ button on your keyboard
- >= greater than or equal
- == equal (comparison)
  - note that = is used for assignment, they're different operators, don't confuse them!
- != not equal

Nested If Statements

You can include conditional statements *inside* of other conditional statements

Such statements are called ***nested*** *if statements*

Describe what will happen,... take it one line at a time

password_nested_ifs.py

```
1  # Alimoor Reza
2  # 09/18/2025
3  # simple nested if else statements
4
5
6  user_name = input("username: ")
7
8  if user_name == "user123":
9      password = input("password: ")
10     if password == "Fall2025":
11         print("Access granted!")
12     else:
13         print("Wrong password!")
14
15 else:
16     print("Incorrect username")
17
```

password_nesteted_ifs.py

Exercise #2

Write the code that will:

Prompt the user for their age

If they are >= 16

    Prompt the user if they've passed the driving exam

    If "yes", print "Congratulations"

    otherwise, print "Sorry. You can't drive"

Otherwise print "you aren't old enough to drive"

Example

Imagine a situation where, given an input, you need to return a grade:

```
Prompt user for their score (between 0 and 100)
If score >= 90, print “you earned an A”
If score >= 80, print “you earned a B”
If score >= 70, print “you earned a C”
If score >= 60, print “you earned a D”
If score < 60, print “you earned an F”
```

Consider this solution

grader_wrong.py

```python
# Alimoor Reza
# 09/18/2025
# simple nested multiple if else statements

score = int(input("Enter your score: "))

if score >= 90:
    print("Your grade is A")
if score >= 80:
    print("Your grade is B")
if score >= 70:
    print("Your grade is C")
if score >= 60:
    print("Your grade is D")
else:
    print("Your grade is F")
```

grader_wrong.py

What happens when I enter 90?

grader_wrong.py

```
1  # Alimoor Reza
2  # 09/18/2025
3  # simple nested multiple if else statements
4
5  score = int(input("Enter your score: "))
6
7  if score >= 90:
8      print("Your grade is A")
9  if score >= 80:
10     print("Your grade is B")
11 if score >= 70:
12     print("Your grade is C")
13 if score >= 60:
14     print("Your grade is D")
15 else:
16     print("Your grade is F")
17
```

Shell

```
>>> %Run grader_wrong.py

  Enter your score: 90
Your grade is A
Your grade is B
Your grade is C
Your grade is D
```

grader_wrong.py

What happens when I enter 70?

grader_wrong.py

```
1  # Alimoor Reza
2  # 09/18/2025
3  # simple nested multiple if else statements
4
5  score = int(input("Enter your score: "))
6
7  if score >= 90:
8      print("Your grade is A")
9  if score >= 80:
10     print("Your grade is B")
11 if score >= 70:
12     print("Your grade is C")
13 if score >= 60:
14     print("Your grade is D")
15 else:
16     print("Your grade is F")
17
```

Shell

```
>>> %Run grader_wrong.py

  Enter your score: 70
  Your grade is C
  Your grade is D

>>>
```

grader_wrong.py

Is this better? Using nested if-else statements?

```
2
3  score = int(input('Enter your score:'))
4
5  if score >= 90:
6      print("Your grade is A")
7  else:
8      if score >= 80:
9          print("Your grade is B")
10     else:
11         if score >= 70:
12             print("Your grade is C")
13         else:
14             if score >= 60:
15                 print("Your grade is D")
16             else:
17                 print("Your grade is F")
18
```

Grader.py

The if-elif-else Statement

if-elif-else statement: special version of a decision structure

Makes logic of nested decision structures simpler to write

Can include multiple `elif` statements

Syntax:

```
if condition1:
    statements
elif condition2:
    statements
else:
    statements
```

grader_if_elif.py

```python
# Alimoor Reza
# 09/18/2025
# simple nested multiple if else statements

score = int(input("Enter your score: "))

if score >= 90:
    print("Your grade is A")
elif score >= 80:
    print("Your grade is B")
elif score >= 70:
    print("Your grade is C")
elif score >= 60:
    print("Your grade is D")
else:
    print("Your grade is F")

```

grader_if_elif.py

What happens when I enter 90?

grader_if_elif.py

```
1  # Alimoor Reza
2  # 09/18/2025
3  # simple nested multiple if else statements
4
5  score = int(input("Enter your score: "))
6
7  if score >= 90:
8      print("Your grade is A")
9  elif score >= 80:
10     print("Your grade is B")
11 elif score >= 70:
12     print("Your grade is C")
13 elif score >= 60:
14     print("Your grade is D")
15 else:
16     print("Your grade is F")
17
```

Shell

```
>>> %Run grader_if_elif.py

  Enter your score: 90
  Your grade is A
```

grader_if_elif.py

What happens when I enter 70?

grader_if_elif.py

```
1  # Alimoor Reza
2  # 09/18/2025
3  # simple nested multiple if else statements
4
5  score = int(input("Enter your score: "))
6
7  if score >= 90:
8      print("Your grade is A")
9  elif score >= 80:
10     print("Your grade is B")
11 elif score >= 70:
12     print("Your grade is C")
13 elif score >= 60:
14     print("Your grade is D")
15 else:
16     print("Your grade is F")
17
```

Shell

```
>>> %Run grader_if_elif.py

  Enter your score: 70
  Your grade is C
```

grader_if_elif.py

Exercise #3

Prompt the user for the number of hours of sleep they got last night.

Use ONLY if-elif-else statements to print out an appropriate response based on the inputted number and the following chart

Above 8	"You are well-rested!"
Between 4 and 8	"The coffee shop is around the corner."
Between 0 and 4	"Are you sure you are awake?"
Less than 0	"input error."

In-Class Exercise

You have ***unlimited*** attempts

Keep on answering until you get all three answers correct

These are indicative of what may appear on a content quiz

What to work on

Lab #2 out today; due on Tuesday, September 23th

Text Reading

https://www.brianheinold.net/python/python_book.html#chapter_ifstatements