Programming Club Meeting 1 Slides

Meeting Goals

- Intro to the club
- Review basic programming and Python
- Practice problem

Intro to the Club

- I am James a senior here
- I am Jon and I am a sophomore here
- The main goal of the club is to get better at programming
- There will be competitions, you don’t have to join but are welcome to
- Your experience in programming isn’t important, the overall goal is just to get better

Intro Cont.

- Expectations:
    - Want to have good time here
    - Still want to focus on actually learning and improving
- We will be meeting on Fridays
    - 1st part - teaching / reviewing things from last week
    - 2nd part - practicing and writing programs

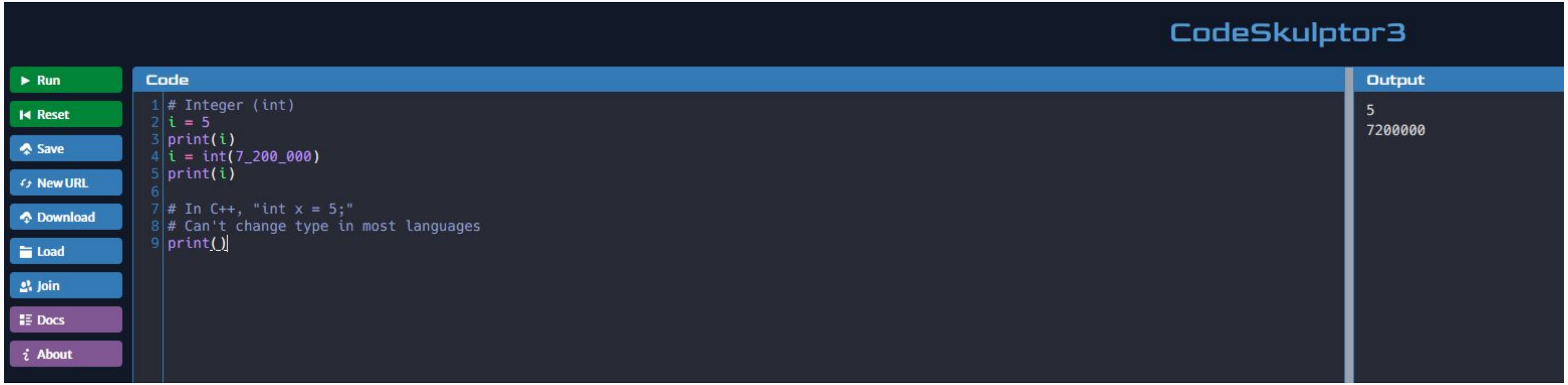
Programming Review Part 1

- How much does everyone know?
- If you don’t have much experience, we’ll be moving a little fast due to people knowing different things
    - If you need help, a better exploration, etc. just say something
    - Feel free to email us with any questions

Programming Review P2

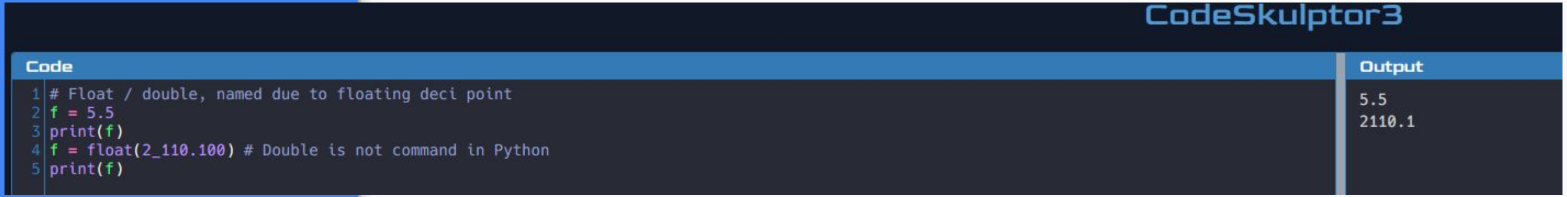
- What is a program?
  - Just a way of solving a problem
- We will be using Python 3
  - Python 2 is very similar
- Where to write code:
  - CodeSkulptor requires no setup and has a big output window
  - VS Code will necessary for competitions and is good for viewing downloaded code (on self service)
    - Good idea to download the code you write
  - Repl.it is good for working with others simultaneously

Programming Review P3 - Basic Data Types



```python
# Integer (int)
i = 5
print(i)
i = int(7_200_000)
print(i)

# In C++, "int x = 5;"
# Can't change type in most languages
print()
```

```
5
7200000
```

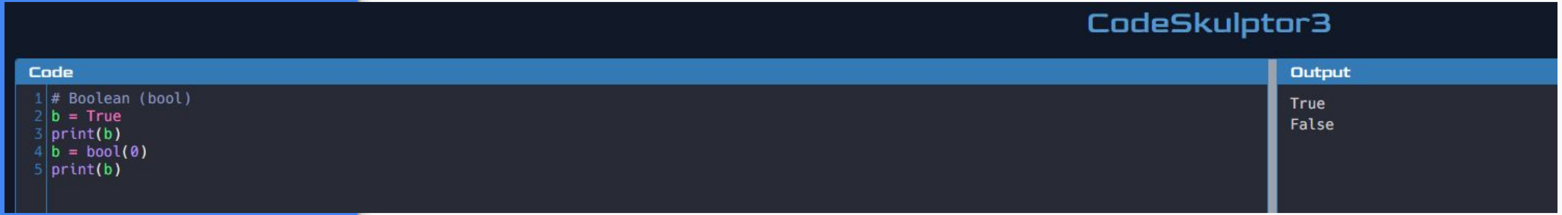


```python
# Float / double, named due to floating deci point
f = 5.5
print(f)
f = float(2_110.100) # Double is not command in Python
print(f)
```

```
5.5
2110.1
```

Floats

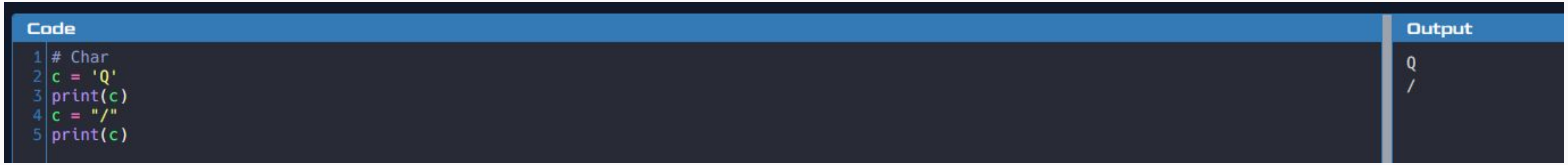
Boolean



```python
# Boolean (bool)
b = True
print(b)
b = bool(0)
print(b)
```

```
True
False
```

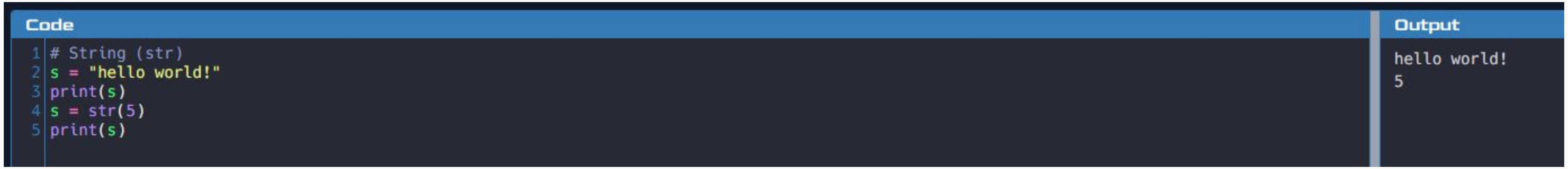
Chars and Strings


Code

```python
# Char
c = 'Q'
print(c)
c = "/"
print(c)
```

Output

```
Q
/
```

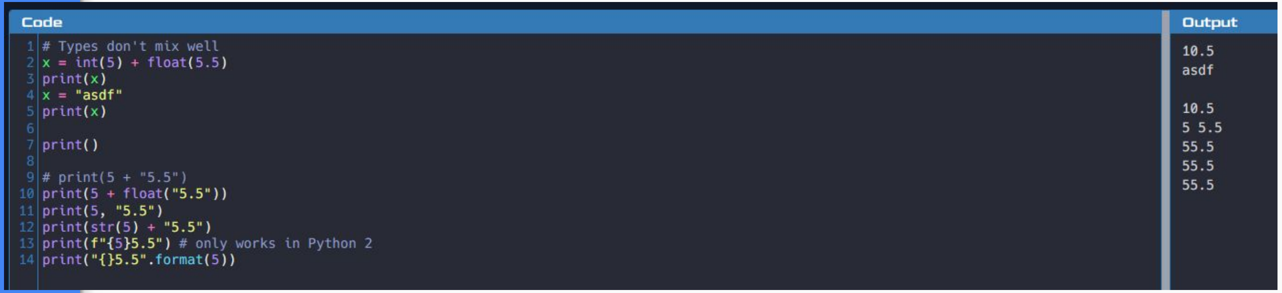

Code

```python
# String (str)
s = "hello world!"
print(s)
s = str(5)
print(s)
```

Output

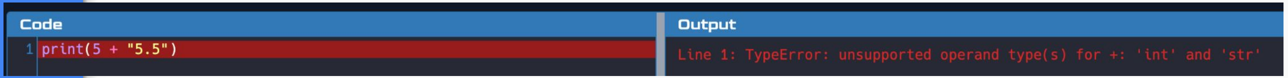
```
hello world!
5
```

Types don’t mix well



```
# Types don't mix well
x = int(5) + float(5.5)
print(x)
x = "asdf"
print(x)

print()

# print(5 + "5.5")
print(5 + float("5.5"))
print(5, "5.5")
print(str(5) + "5.5")
print(f"{5}5.5") # only works in Python 2
print("{}5.5".format(5))
```

```
10.5
asdf

10.5
5 5.5
55.5
55.5
55.5
```

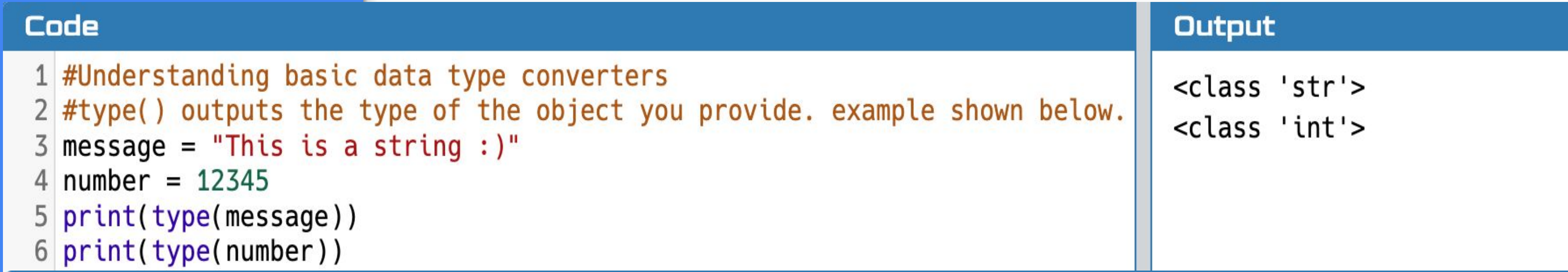


```
print(5 + "5.5")
```

```
Line 1: TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

Data Type Converters

type()

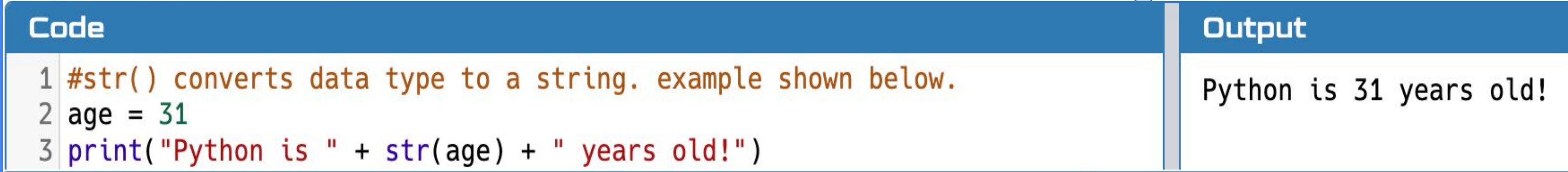

Code

```
#Understanding basic data type converters
#type( ) outputs the type of the object you provide. example shown below.
message = "This is a string :)"
number = 12345
print(type(message))
print(type(number))
```

Output

```
<class 'str'>
<class 'int'>
```

str()

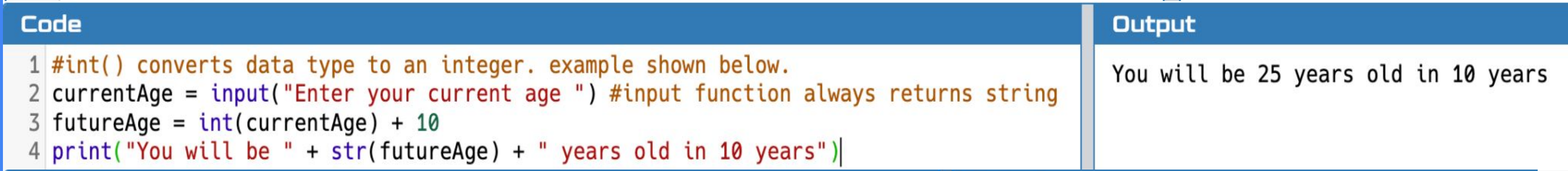

Code

```
#str( ) converts data type to a string. example shown below.
age = 31
print("Python is " + str(age) + " years old!")
```

Output

```
Python is 31 years old!
```

str()

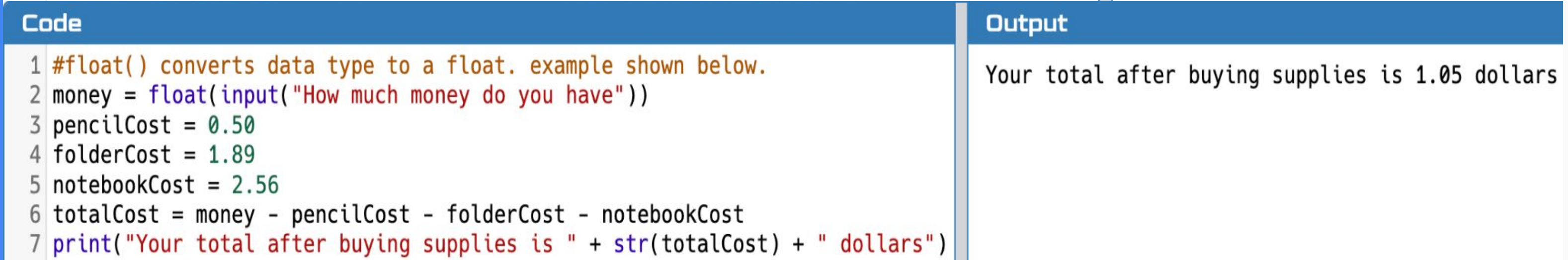

Code

```
#int( ) converts data type to an integer. example shown below.
currentAge = input("Enter your current age ") #input function always returns string
futureAge = int(currentAge) + 10
print("You will be " + str(futureAge) + " years old in 10 years")
```

Output

```
You will be 25 years old in 10 years
```

float()


Code

```
#float( ) converts data type to a float. example shown below.
money = float(input("How much money do you have"))
pencilCost = 0.50
folderCost = 1.89
notebookCost = 2.56
totalCost = money - pencilCost - folderCost - notebookCost
print("Your total after buying supplies is " + str(totalCost) + " dollars")
```

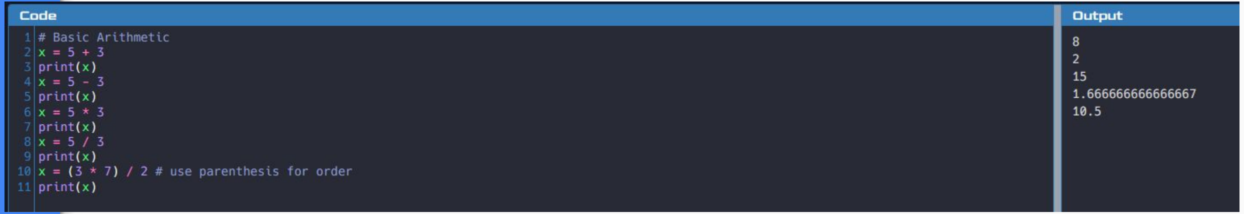
Output

```
Your total after buying supplies is 1.05 dollars
```

Programming Review P4 - Var Names

- Variable names should be descriptive
  - Don’t use just “x” or “y” like I’ve been in the rest of this powerpoint
- Should use camel case or underscores
  - Variable for the number of people
    - numOfPeople
    - num_of_people
  - Can use either
- Bools have a weird style of names: isFinished

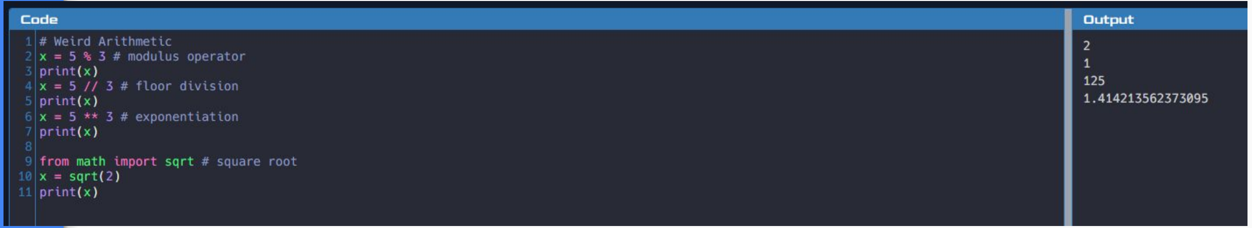
Programming Review P5 - Basic Commands


Code

```
# Basic Arithmetic
x = 5 + 3
print(x)
x = 5 - 3
print(x)
x = 5 * 3
print(x)
x = 5 / 3
print(x)
x = (3 * 7) / 2 # use parenthesis for order
print(x)
```

Output

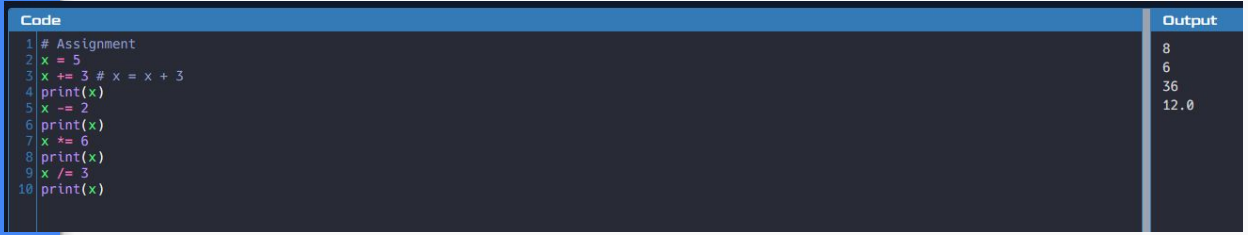
```
8
2
15
1.6666666666666667
10.5
```


Code

```
# Weird Arithmetic
x = 5 % 3 # modulus operator
print(x)
x = 5 // 3 # floor division
print(x)
x = 5 ** 3 # exponentiation
print(x)

from math import sqrt # square root
x = sqrt(2)
print(x)
```

Output

```
2
1
125
1.4142135623730951
```

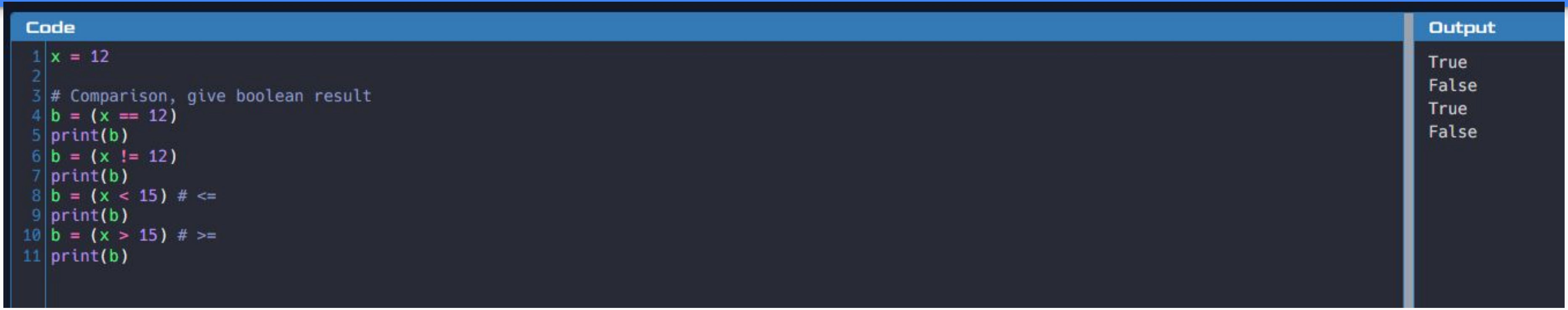

Code

```
# Assignment
x = 5
x += 3 # x = x + 3
print(x)
x -= 2
print(x)
x *= 6
print(x)
x /= 3
print(x)
```

Output

```
8
6
36
12.0
```

Comparison

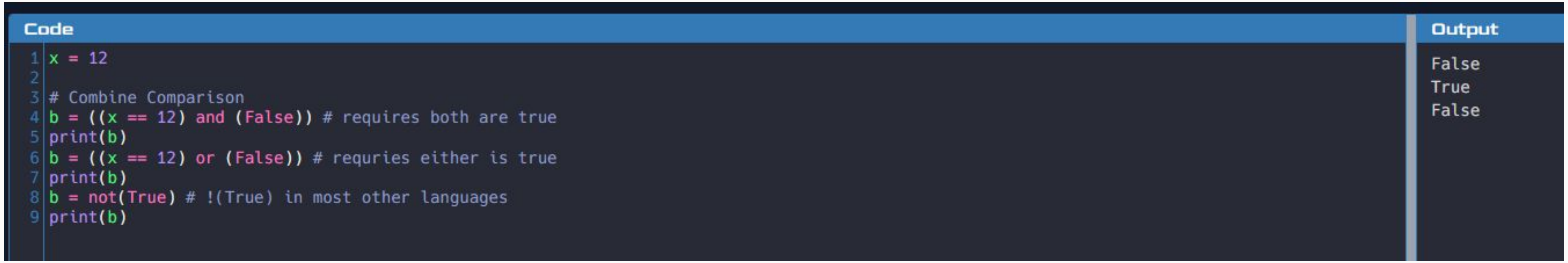


Code

```
x = 12

# Comparison, give boolean result
b = (x == 12)
print(b)
b = (x != 12)
print(b)
b = (x < 15) # <=
print(b)
b = (x > 15) # >=
print(b)
```

Output

```
True
False
True
False
```



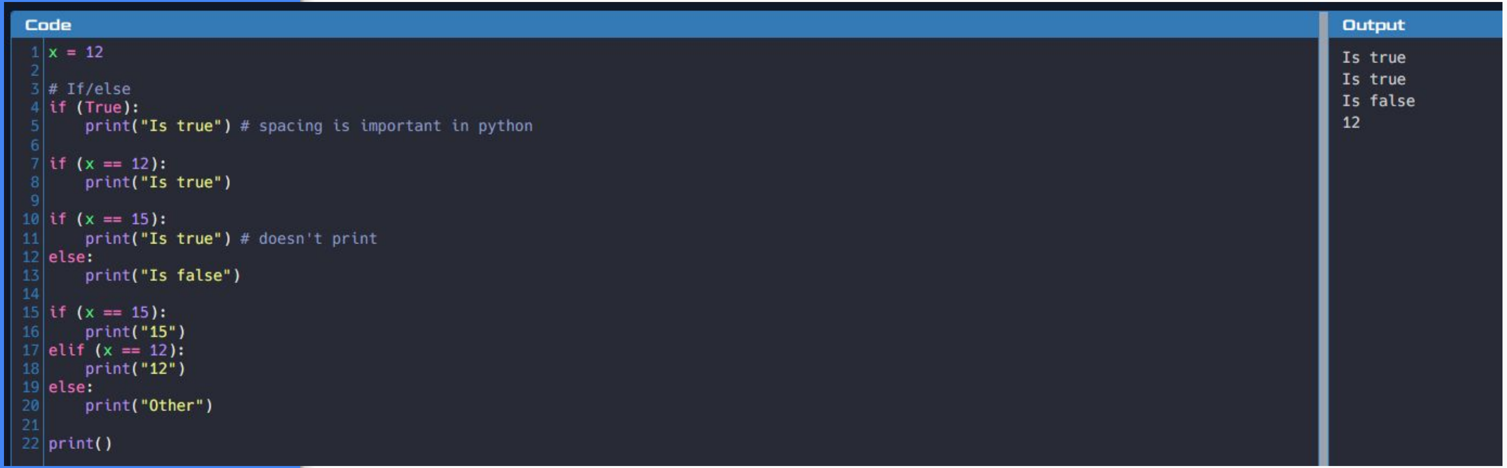
Code

```
x = 12

# Combine Comparison
b = ((x == 12) and (False)) # requires both are true
print(b)
b = ((x == 12) or (False)) # requries either is true
print(b)
b = not(True) # !(True) in most other languages
print(b)
```

Output

```
False
True
False
```

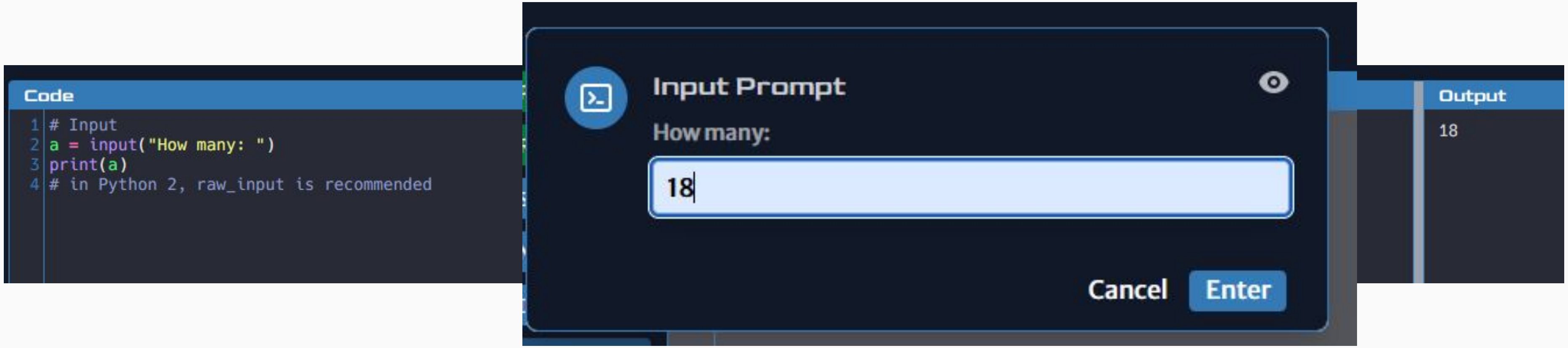
If/else



```python
x = 12

# If/else
if (True):
    print("Is true") # spacing is important in python

if (x == 12):
    print("Is true")

if (x == 15):
    print("Is true") # doesn't print
else:
    print("Is false")

if (x == 15):
    print("15")
elif (x == 12):
    print("12")
else:
    print("Other")

print()
```

```
Is true
Is true
Is false
12
```

Input (always returns a string data type)



```
# Input
a = input("How many: ")
print(a)
# in Python 2, raw_input is recommended
```

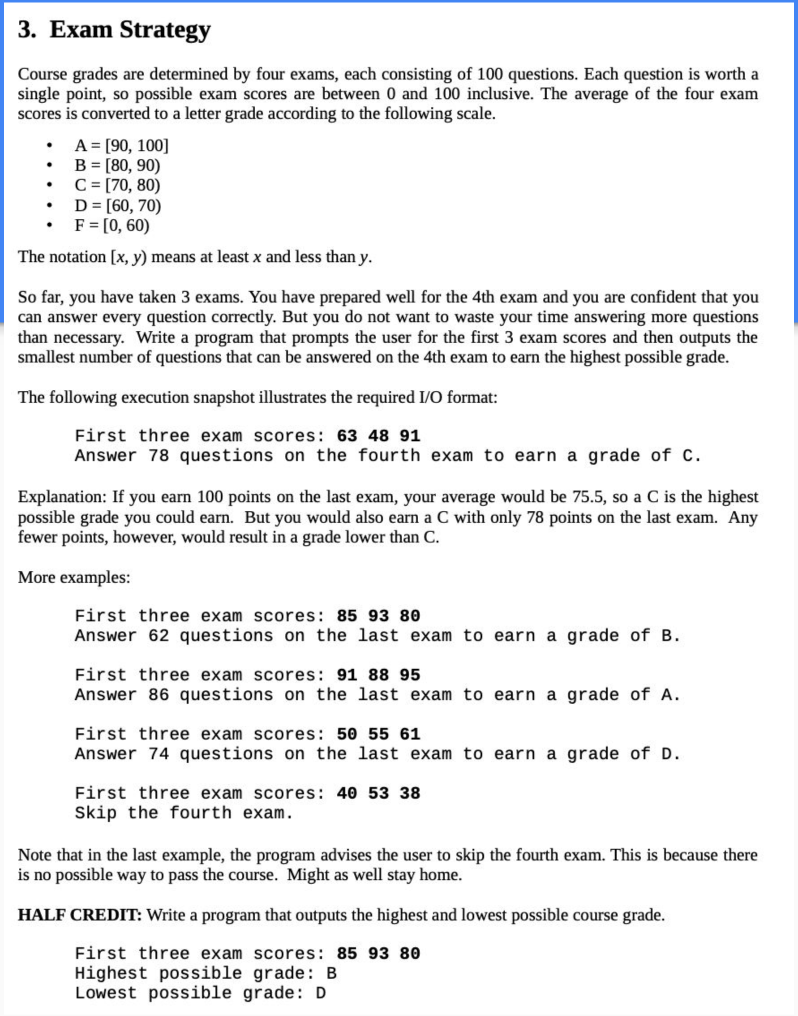
www.w3schools.com is a great reference site

Practice Problem 1 - If the Earth was made of Legos

- Src: https://www.101computing.net/what-if-planet-earth-was-made-of-lego/
- Goal: Write a program to determine how many lego bricks would be needed to build planet Earth.
- Relevant Information:
  - Radius of the Earth is ~6731 km
  - The dimensions of a Lego bricks are:
    - L = 16 mm
    - W = 16 mm
    - H = 10 mm
  - 1 km = 1,000,000 mm
  - Volume of a sphere = (4/3) * pi * r^2
  - Volume of a brick = L * W * H
- Notes:
  - The final answer is very large (Python will output it in scientific notation)
  - This problem is far easier if you convert the radius of the earth into mm early on

Practice Problem 2 - Exam Strategy

Can put the inputs on 3 different lines instead of just 1



3. Exam Strategy

Course grades are determined by four exams, each consisting of 100 questions. Each question is worth a single point, so possible exam scores are between 0 and 100 inclusive. The average of the four exam scores is converted to a letter grade according to the following scale.

- A = [90, 100]
- B = [80, 90)
- C = [70, 80)
- D = [60, 70)
- F = [0, 60)

The notation [*x*, *y*) means at least *x* and less than *y*.

So far, you have taken 3 exams. You have prepared well for the 4th exam and you are confident that you can answer every question correctly. But you do not want to waste your time answering more questions than necessary. Write a program that prompts the user for the first 3 exam scores and then outputs the smallest number of questions that can be answered on the 4th exam to earn the highest possible grade.

The following execution snapshot illustrates the required I/O format:

```
First three exam scores: 63 48 91
Answer 78 questions on the fourth exam to earn a grade of C.
```

Explanation: If you earn 100 points on the last exam, your average would be 75.5, so a C is the highest possible grade you could earn. But you would also earn a C with only 78 points on the last exam. Any fewer points, however, would result in a grade lower than C.

More examples:

```
First three exam scores: 85 93 80
Answer 62 questions on the last exam to earn a grade of B.

First three exam scores: 91 88 95
Answer 86 questions on the last exam to earn a grade of A.

First three exam scores: 50 55 61
Answer 74 questions on the last exam to earn a grade of D.

First three exam scores: 40 53 38
Skip the fourth exam.
```

Note that in the last example, the program advises the user to skip the fourth exam. This is because there is no possible way to pass the course. Might as well stay home.

HALF CREDIT: Write a program that outputs the highest and lowest possible course grade.

```
First three exam scores: 85 93 80
Highest possible grade: B
Lowest possible grade: D
```

- If anyone wants to continue working or get feedback, you can ask either of us
- There will be a GitHub (https://github.com/battler82004/jt-prgrm-club-22-23) where we'll post solutions and the notes from the lesson
    - Will hopefully uploads solutions on Thursdays
    - To contact James: James.Taddei@jtasd.org
    - To contact Jon: Jonathan.Duffy@jtasd.org
- Next week:
    - Hoping to look at loops and string manipulation next week

Practice Problem Cont.

Upcoming Coding Competition

- When? **Saturday, October 22, 8am - 1pm**
- Where? **DeSales University**
- Hosted By? **ACM, SIGCSE, and DeSales University.**
- **More Information:** DeSales University is hosting an undergraduate computer science conference. The conference includes a programming competition and High School students are welcome to attend. The competition will be held on Saturday, Oct 22nd from 8-1pm.

CONTACT MRS. GABRIELLE IF INTERESTED IN ATTENDING