Introduction to Programming

Alexander Hom

March 7, 2020

Association for Computing Machinery (ACM)

Introduction

What is Programming?

Today, computers are very smart and powerful, but we often want them to do a specific task that they might not be able to perform on their own. That is where programming comes in. Programming allows us to *talk* with a computer and let it know what we want it to do.

How does Programming Work?

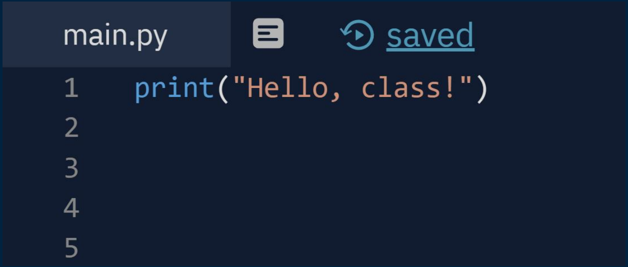
Programming sounds *interesting*...but how do we do it?

In order for us to communicate with the computer, we need to use a language that both we and the computer can understand. This language takes the form of code.

Code Example

`print("Hello, Class!")` $\xrightarrow{\text{translates into}}$ Hello, Class!

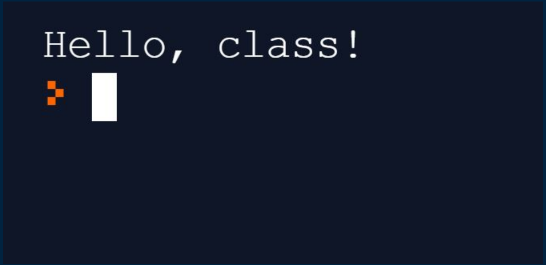
Code Editor



```python
print("Hello, class!")
```

Code is written into the code editor

Code Output



```
Hello, class!
```

The result of our code is displayed in the console

Cool Programming Applications

Even though we might not see them directly (since code is hidden behind the scenes), the applications of programming can be found everywhere. Some examples include:

- The apps on our smartphones and computers.
- Websites.
- Computers.
- Even this presentation.

What are We Learning Today?

Today, we are going to be learning the basics of a programming language called Python. We are going to be writing our code in an online coding environment, which makes it easy for us to run and see the results of our code.

What is Python?

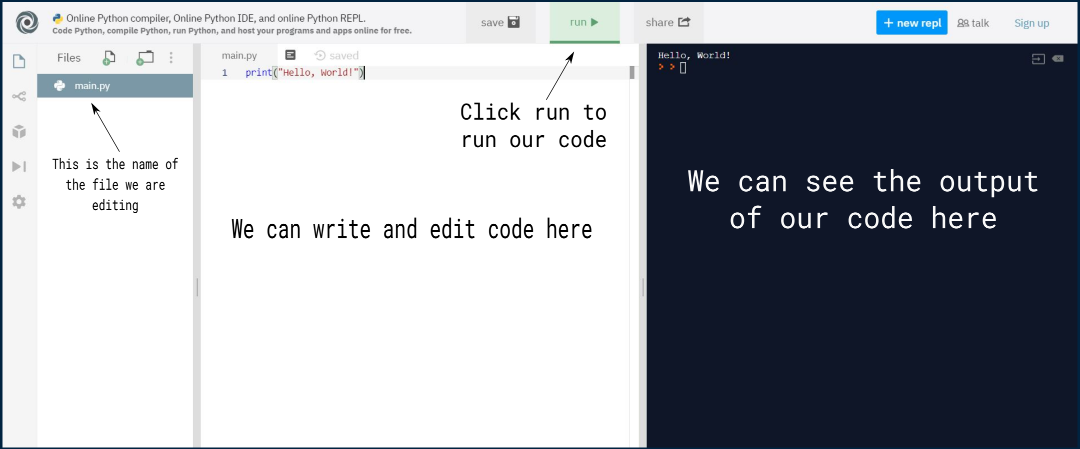
Python is a popular programming language and one of the best first languages to learn. It is very approachable to beginners compared to other languages because Python code is read like English sentences and has a simpler syntax (guideline that the code needs to follow).

Activities

Getting Started

Before we start writing Python code, let's get familiar with the environment with which we are coding on.

The Coding Environment



Online Python compiler, Online Python IDE, and online Python REPL.

Code Python, compile Python, run Python, and host your programs and apps online for free.

```
print("Hello, World!")
```

```
Hello, World!
> > 
```

This is the name of the file we are editing

Click run to run our code

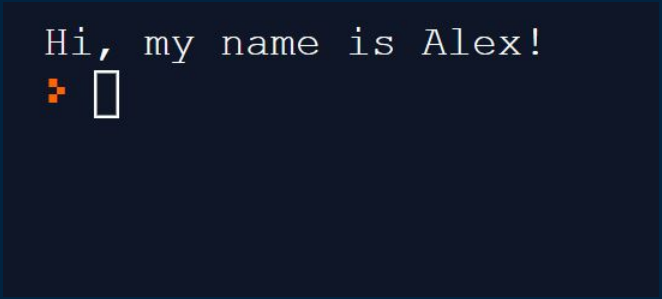
We can write and edit code here

We can see the output of our code here

Activity 1: Hello World Program

Let’s start off the coding activities by coding the classic *Hello World* program.

Activity 2: Print Statements



```
Hi, my name is Alex!
>
```

Let’s try modifying the code so that it prints something else, like your name for instance.

Introduction to Python Grammar Rules (Syntax)

Just like in English, there is a gramatical structure that we have to follow when programming in Python and other languages. This structure is called the code's syntax. The syntax for Python's `print` function goes like this:

$$\underbrace{\texttt{print}}_{\text{Action}}\texttt{(}\ \underbrace{\texttt{"Hi, my name is Alex!"}}_{\text{Object that the action is being done on}}\ \texttt{)}$$

There are many other gramatical rules in Python that programmers must follow, which I will introduce later on.

Data Types

In the previous two activities, we printed out words. However, we can print out other types of data as well. Here are some *data types* that you will encounter in Python and other programming languages:

1. `string` → Words/Phrases/Sentences (i.e., "Hello", "I am a string", . . .)
2. `char` → Characters (i.e., 'A', 'B', 'C', . . .)
3. `int` → Non-decimal numbers (integers) (can be positive or negative)
4. `double` → Decimal numbers (can be positive or negative)
5. `bool` → True (1) or false (0)

Performing Math Operations

So far, we've printed `strings` to the console, but we can also print out other data types, such as numbers (`int` and `double`). One of the nice things about Python is that we can perform math operations on `ints` and `doubles` and print out the results to the console:

Addition: `print(5 + 10)` → Output: 15

Subtraction: `print(8.0 - 2.0)` → Output: 6.0

Multiplication: `print(17 * 3)` → Output: 51

Division: `print(30.0 / 2.0)` → Output: 15.0

Activity 3: Printing Numbers

Pick two random numbers (for instance, 288 and 97):

1. Print out the sum of the two numbers
2. Print out the product of the two numbers

Introduction to Variables

We just printed out the sum and product of two numbers. But, what if we want to save the results and use them later on? That's where variables come into play.

Variables are *names* that we can assign to data. For instance, we can add together the numbers 288 and 97 and assign the total number to the variable `total`.

```
total = 288 + 97
```

Rules of Variables

To use variables in our code, we have to first introduce them to the computer, otherwise the computer won't know what the variable is:

Introducing the variables:

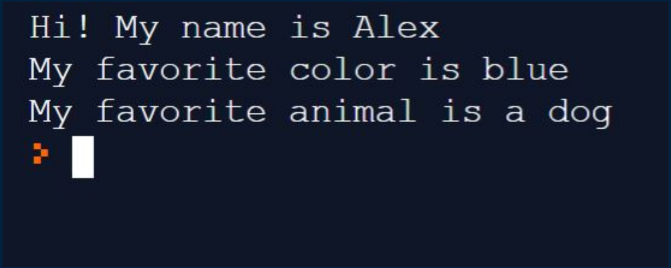
- `product = 56 * 25`
- `name = "Alex"`

Calling the variables:

- print(product) → Output: 1400
- print(”My name is” + name) → Output: My name is Alex

Activity 4: Variables Practice

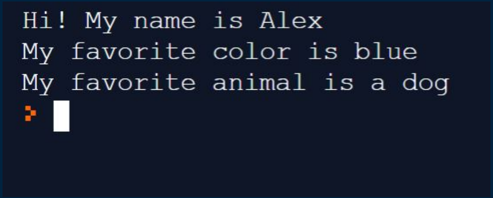
Let's use what we have learned so far about printing out words and variables to write a program that displays our name, our favorite color, and our favorite animal.



```
Hi! My name is Alex
My favorite color is blue
My favorite animal is a dog
> █
```

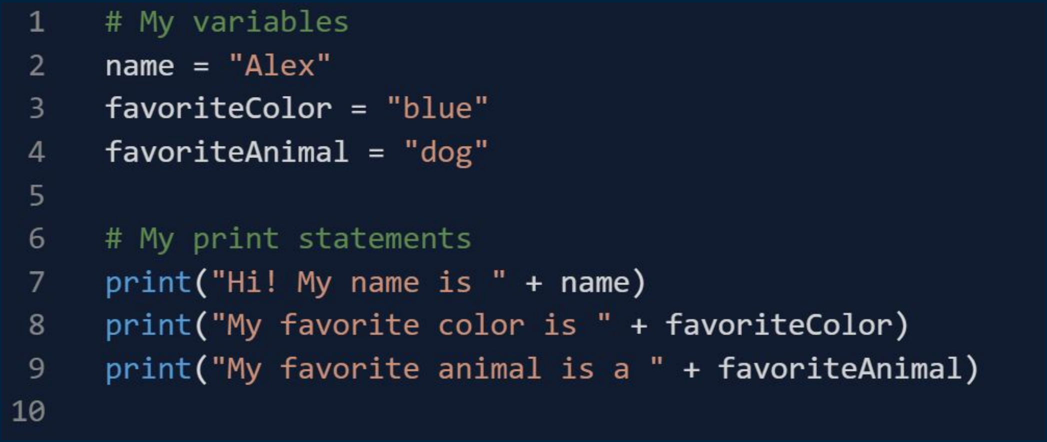
Activity 4: Tips and Hints

- Declare all your variables first
- Use simple names for your variables, like `favoriteColor`
- Variables can't have spaces. If your variable has more than one word, use:
    - Camel case → `favoriteColor`
    - Hyphen → `favorite-color`
    - Underscore → `favorite_color`



```
Hi! My name is Alex
My favorite color is blue
My favorite animal is a dog
›
```

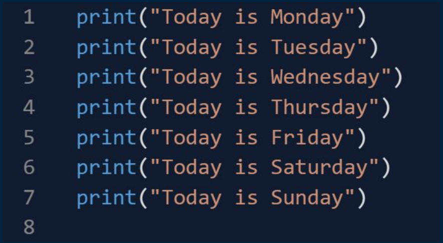
Activity 4: Example Code



```python
# My variables
name = "Alex"
favoriteColor = "blue"
favoriteAnimal = "dog"

# My print statements
print("Hi! My name is " + name)
print("My favorite color is " + favoriteColor)
print("My favorite animal is a " + favoriteAnimal)
```

Adding Logic to our Code

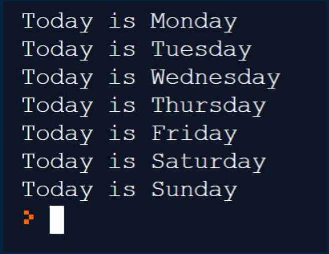
So far, our programs have been one-dimensional, meaning that they run every line of code and perform every action. But what if we want to run certain lines depending on the circumstance, which can change. For example, we have a program that prints the days of the week:



```python
print("Today is Monday")
print("Today is Tuesday")
print("Today is Wednesday")
print("Today is Thursday")
print("Today is Friday")
print("Today is Saturday")
print("Today is Sunday")

```

Prints out every day of the week

→



```
Today is Monday
Today is Tuesday
Today is Wednesday
Today is Thursday
Today is Friday
Today is Saturday
Today is Sunday
```

Only want to print out Saturday

Introduction to Conditional Statements

Rather than having the program print out all the days of the week, we can have it just print out today, Saturday, with the help of a conditional statement.

In English, conditionals are the ifs and elses that we sometimes use in our conversations:

If it is raining, then I will bring an umbrella. Otherwise (else), I won't.

If today is Saturday, then the program will print out Saturday. Else, it will print out a different day.

The If-Else statement

To add a condition statement in our programs, we can use if-else statements. The syntax for if-else statments in Python goes like this:

```
if (condition_1):
  perform action_1
elif (condition_2):
  perform action_2
else:
  perform action_3
```

If-Else Example

```
today = "Saturday"

if (today == "Saturday"):
  print("Today is Saturday")
else:
  print("Today is NOT Saturday")
```

→

```
Today is Saturday
```

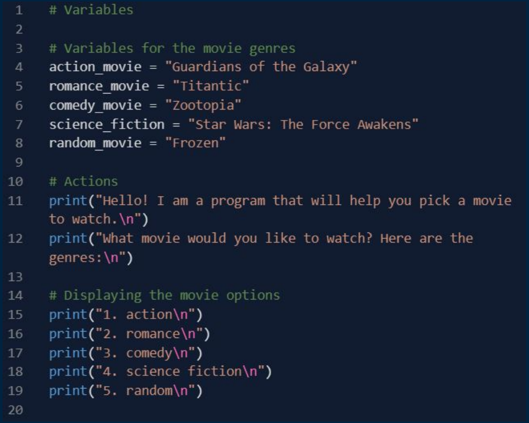
Activity 5: If-Else Statements Practice

Sam is planning on watching a movie tonight, but is unsure of what movie to watch. Let's write a program using if-else statements that will help Sam decide on a movie.

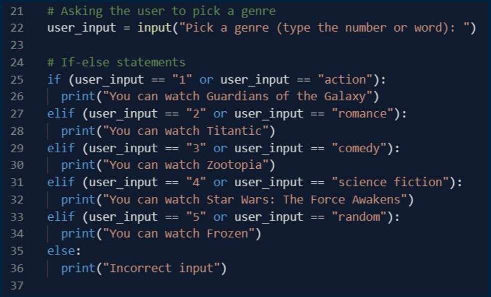
Activity 5: Outline of Movie-Choosing Program

- Sam can choose from five genres of movies (action, romance, comedy, science fiction, and random)
- The program will list the movie genres and ask Sam to pick from one of the five genres
- The program will display a movie based on which genre Sam picked

Activity 5: Code



```python
# Variables

# Variables for the movie genres
action_movie = "Guardians of the Galaxy"
romance_movie = "Titantic"
comedy_movie = "Zootopia"
science_fiction = "Star Wars: The Force Awakens"
random_movie = "Frozen"

# Actions
print("Hello! I am a program that will help you pick a movie to watch.\n")
print("What movie would you like to watch? Here are the genres:\n")

# Displaying the movie options
print("1. action\n")
print("2. romance\n")
print("3. comedy\n")
print("4. science fiction\n")
print("5. random\n")

```

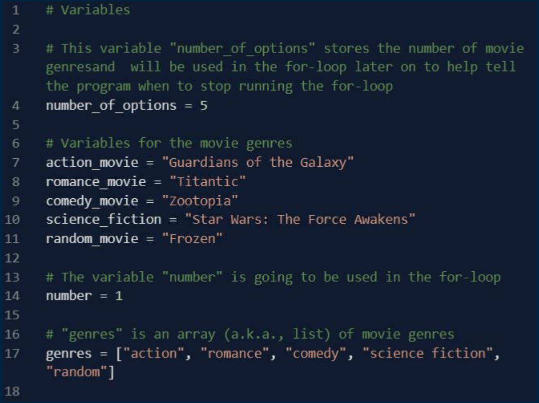


```python
# Asking the user to pick a genre
user_input = input("Pick a genre (type the number or word): ")

# If-else statements
if (user_input == "1" or user_input == "action"):
  print("You can watch Guardians of the Galaxy")
elif (user_input == "2" or user_input == "romance"):
  print("You can watch Titantic")
elif (user_input == "3" or user_input == "comedy"):
  print("You can watch Zootopia")
elif (user_input == "4" or user_input == "science fiction"):
  print("You can watch Star Wars: The Force Awakens")
elif (user_input == "5" or user_input == "random"):
  print("You can watch Frozen")
else:
  print("Incorrect input")

```

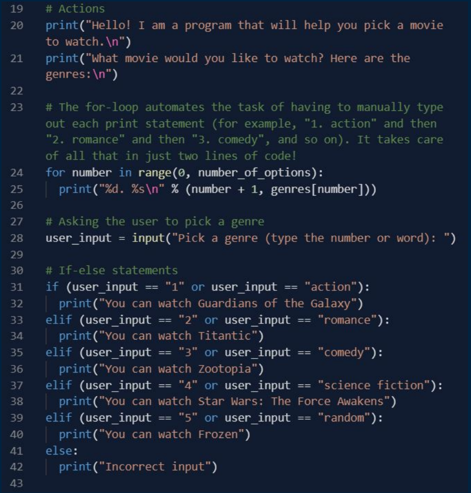
Activity 5 Bonus: Alternate Movie-Choosing Program

As a bonus: here's another way of performing the same thing as the previous program, but this time using something called `arrays` and `for-loops` that help make the code more efficient.

Activity 5 Bonus: Code

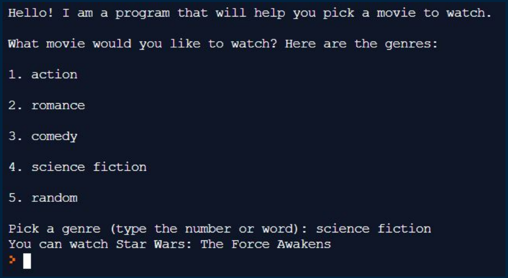


```python
# Variables

# This variable "number_of_options" stores the number of movie genresand  will be used in the for-loop later on to help tell the program when to stop running the for-loop
number_of_options = 5

# Variables for the movie genres
action_movie = "Guardians of the Galaxy"
romance_movie = "Titantic"
comedy_movie = "Zootopia"
science_fiction = "Star Wars: The Force Awakens"
random_movie = "Frozen"

# The variable "number" is going to be used in the for-loop
number = 1

# "genres" is an array (a.k.a., list) of movie genres
genres = ["action", "romance", "comedy", "science fiction", "random"]

```

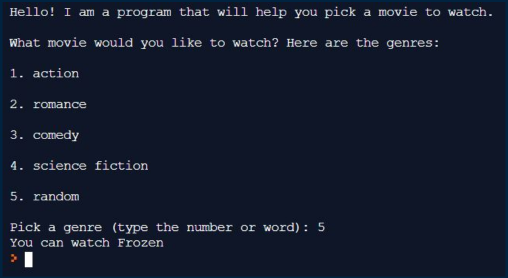


```python
# Actions
print("Hello! I am a program that will help you pick a movie to watch.\n")
print("What movie would you like to watch? Here are the genres:\n")

# The for-loop automates the task of having to manually type out each print statement (for example, "1. action" and then "2. romance" and then "3. comedy", and so on). It takes care of all that in just two lines of code!
for number in range(0, number_of_options):
    print("%d. %s\n" % (number + 1, genres[number]))

# Asking the user to pick a genre
user_input = input("Pick a genre (type the number or word): ")

# If-else statements
if (user_input == "1" or user_input == "action"):
    print("You can watch Guardians of the Galaxy")
elif (user_input == "2" or user_input == "romance"):
    print("You can watch Titantic")
elif (user_input == "3" or user_input == "comedy"):
    print("You can watch Zootopia")
elif (user_input == "4" or user_input == "science fiction"):
    print("You can watch Star Wars: The Force Awakens")
elif (user_input == "5" or user_input == "random"):
    print("You can watch Frozen")
else:
    print("Incorrect input")

```

Activity 5: Output



```
Hello! I am a program that will help you pick a movie to watch.

What movie would you like to watch? Here are the genres:

1. action

2. romance

3. comedy

4. science fiction

5. random

Pick a genre (type the number or word): science fiction
You can watch Star Wars: The Force Awakens
›
```



```
Hello! I am a program that will help you pick a movie to watch.

What movie would you like to watch? Here are the genres:

1. action

2. romance

3. comedy

4. science fiction

5. random

Pick a genre (type the number or word): 5
You can watch Frozen
›
```

Both versions of the program display the same output

Wrapping Up

Recap of the Workshop

In today's workshop we learned:

1. What programming is and some of its real-world applications
2. How to interact with an online code editor called repl.it
3. How to print words and numbers in Python
4. Some grammar rules (syntax) that applies to Python
5. How to perform math operations using Python
6. The basics of variables
7. How to implement some logic in our code using if-else statements

Questions?

Thank You!