# Unleashing the Superpowers of Functions: An Experiential and Active Learning Journey

ENDG 233 – Programming with Data

University of Calgary

The Schulich School of Engineering (SSE)

Shadi Aljendi & TA

June 27, 2023

## Learning Objectives

After finishing this activity, you will be able to

1. Comprehend the concept and significance of programming functions.
2. Acquire the skills to declare, define, and utilize functions in Python.
3. Attain expertise in modular programming and code organization using functions.
4. Apply functions proficiently to resolve programming problems with effectiveness and efficiency.
5. Foster code reusability and maintainability through the strategic implementation of functions.

## General Instructions

1. Engage in experiential and active learning activities to reinforce your understanding of functions in programming.
2. Work individually or in small groups, depending on the activity instructions.
3. Participate actively in discussions, problem-solving, and hands-on coding exercises.

Utilize the provided code and resources from the GitHub repository to enhance your learning experience. Use your preferred EDI to work on the code. If you prefer, you can use the paper version to complete your work. The link to access the resources is GitHub Repository: https://github.com/shadialjendi/PythonFunction/. Here you can find to folders:

- The Code Folder: Contains the code for the pre-recorded lecture.
- The Activity Folder: Contains the skeleton program files for the activities. These files provide a starting point with code structures that you will need to complete. To access the activity files, click on the "Activity" folder and choose the corresponding file for the activity you are working on.

Please download the necessary files and code snippets from the repository to follow along with the activities. If you have any questions or encounter any issues, don't hesitate to ask for assistance from your peers, the TA, or the instructor.

4. Each activity is designed to be completed within a specific time frame. Please manage your time effectively to ensure the completion of all activities.
5. The TA as well as the instructor are available to answer any questions or address any challenges you may encounter during the activities. Don't hesitate to ask your questions publicly, if you think they can be beneficial for the entire class. Additionally, paying attention to the questions asked by others can save time and provide additional insights for everyone involved.
6. At the end of each activity, the correct code will be provided for reference and comparison.
7. The main workload of this session is the three first activities (Activity 1, Activity 2 and Activity 3) that you are expected to finish during ~20 minutes. **If you finish early**, we are challenging you to do the Extra Challenge Activity, or at least start working on it.

## Activity 1: Fill in the Blanks ~(5 minutes)

It is recommended that you do this activity individually.

In the Activity folder of the provided GitHub repository, you will find the `Temperatures.py` file. This file includes a program that performs temperature conversions between Celsius and Fahrenheit. The program utilizes two functions: `celsius_to_fahrenheit(celsius)` and `fahrenheit_to_celsius(fahrenheit)`. These functions are called from the main program. However, the program has **eight blanks** that need to be filled for this activity. The blanks are denoted as "_____" (without the ""), and are accompanied by a comment and a number at the end of the respective line.

If you prefer carrying out the activity on paper, here is the code:

```
_________________ celsius_to_fahrenheit(celsius): #1
    fahrenheit = (celsius * 9/5) + 32
    _________________ fahrenheit #2


_________________ fahrenheit_to_celsius(fahrenheit): #3
    celsius = (fahrenheit - 32) * 5/9
    _________________ celsius #4


temperature = float(input("Enter the temperature: "))
unit = input("Enter the unit (C for Celsius, F for Fahrenheit): ")
```

```
_________________ unit == 'C': #5
    converted_temp = _________________(temperature) #6
else :
    converted_temp = _________________(temperature) #7

print("The converted temperature is", _________________) #8
```

Once you have completed filling in the blanks, you can test and debug your program within your preferred Integrated Development Environment (IDE).

## Activity 2: Create a function ~(5 minutes or 20 person/minutes)

It is recommended that you carry out this activity in groups of four with the students at your table.

In Activity 1, we have used two functions to do the conversion between the temperature units. The program tested the read unit and accordingly called one of the functions: either `celsius_to_fahrenheit(celsius)` or `fahrenheit_to_celsius(fahrenheit)`.

In this activity, you are required to use only one function to carry out the same task. The starting file in GitHub that you can use for this activity is `Temperatures2.py`. Use the file to complete the function `convert_temperature(temp, unit)`.

If you prefer carrying out the activity on paper, here is the code:

```
def convert_temperature(temp, unit):
```

```
# Main program
temperature = float(input("Enter the temperature: "))
unit = input("Enter the unit (C for Celsius, F for Fahrenheit): ")

print(convert_temperature(temperature, unit))
```

Once you have completed the function, you can test and debug your program within your preferred Integrated Development Environment (IDE).

## Activity 3: Using functions to create a modular program ~(10 minutes or 20 person/minutes)

It is recommended that you carry out this activity in groups of two with the students at your table.

The code in the GitHub file OddEven0.py contains a non-modular program that uses a `for` loop to read five integers and count the number of even and odd integers and print these numbers at the end. Modify this program to be modular by creating a function that tests if the entered integer is even or odd and use this function in your program.

If you prefer carrying out the activity on paper, you can use the space after the OddEven0.py program to write your own code.

```
even_count = 0
odd_count = 0


for _ in range(5):
    num = int(input("Enter a number: "))
    if num % 2 == 0:
        even_count += 1
    else:
        odd_count += 1


print("Number of even numbers:", even_count)
print("Number of odd numbers:", odd_count)
```

Here is the space you can use **if you prefer to complete the work on paper**.

Once you have completed the function, you can test and debug your program within your preferred Integrated Development Environment (IDE).

## Extra Challenge Activity: Develop a modular function from scratch

For this Extra Challenge Activity, you have the flexibility to choose whether you want to work individually or in groups. However, if you decide to work in a group, please collaborate with students at your table.

The objective of this activity is to write a Python program that utilizes at least one function. The program should only print the composite Fibonacci numbers that are smaller than a specified number, which will be read as input.

The following is a sample output of the requested program.

```
Enter the limit for Fibonacci numbers: 20
0
1
1
8
```

Once you have completed the function, you can test and debug your program within your preferred Integrated Development Environment (IDE).

## Conclusion

Congratulations on completing the activities for this Flipped Learning Module on Functions in Programming! Throughout the module, you have gained a deeper understanding of the concept and

importance of functions, learned how to declare and define functions in Python, and applied functions to solve programming problems effectively.

By actively participating in the provided activities, you have had the opportunity to enhance your coding skills and experience firsthand the benefits of modular programming and code organization using functions. Through the collaborative learning environment, you were able to engage in discussions, ask questions, and seek assistance when needed.

Remember, the use of functions in programming not only improves code reusability and maintainability but also allows for efficient problem-solving. By mastering the use of functions, you have taken a significant step towards becoming a proficient programmer.

We encourage you to continue exploring the world of programming and further apply your newfound knowledge and skills. Practice implementing functions in your own projects and seek out additional resources to expand your understanding of programming concepts.

Thank you for your active participation and dedication throughout this module. We hope you have found it valuable in your programming journey. If you have any further questions or require additional support, feel free to reach out to the course instructor or TA.

Best wishes on your continued programming endeavours!

The ENDG 233 Instruction Team