**Download Your Daily Notes**

**Daily Notes - Input and Output**

 Functions are essential in programming when a specific operation needs to be repeated with slight variations. Instead of duplicating the code, functions provide a more efficient solution. They are reusable blocks of code that can be called whenever the operation needs to be performed. Functions can take zero or multiple inputs, known as parameters.

**Daily Notes - Reading and Writing Files**

 Functions are created by providing a name, a list of parameters (optional), and a block of code that may return a value. Functions are typically defined in separate script files to avoid duplication. It's important to distinguish between arguments (used in function applications) and parameters (local variables within the function). The syntax for defining a function is "def function\_name(parameter <,...>): #suite". The first line of a function should be a comment explaining its purpose. Functions can have a return statement to provide a result or break out of the function.  
  
There are three types of functions in Python: ordinary functions (perform calculations and return results), procedure functions (execute a specific procedure), and factory functions (generate values without parameters). An example was provided to illustrate how functions improve efficiency in programs.  
  
In the example, a function called calculateTax was defined to calculate and print the tax based on a given salary. The program is divided into two parts: the main program and the calculateTax function. Local variables within the function, such as rate and tax, are only accessible within the function itself. To make a local variable's value available outside the function, it can be returned using a return statement.  
  
The concept of default parameters was also explained. By assigning a default value to a parameter in a function's declaration, that value is automatically used if the parameter is not explicitly provided in a function call. An example was given, showing how changing the value of a variable outside the function does not affect the default value of a parameter.

**Daily Notes - Activity 1 - Read Methods**

 This has been completed and uploaded. It basically asks the user for temperature input and then converts it into degrees Celsius. It the also displays which day was colder.

**Daily Notes - Write Methods**

 This covers the random module in Python, which provides functions for generating random numbers. The module utilizes the Mersenne Twister algorithm, known for its long period and fast performance. The random module functions are bound methods of the random.Random class.  
  
An example program was provided, demonstrating the use of the random module. The program generates random lottery numbers using the randrange function. However, it contains a logical error as it doesn't check for duplicate numbers. The program's flaw will be addressed later.  
  
The random module offers other methods, including randint (generates a random integer within a specified range), sample (selects unique random elements from a population), and choice (selects a random element from a sequence).  
  
Another example program was given, prompting the user to input three numbers. These numbers are passed to two functions: determineProduct (calculates and prints their product) and determineAve (calculates and returns their average). The average is then printed in the main program.  
  
Key points to note in the example program are: both functions are called sequentially in the main program, the first function calculates and prints the product, and the second function calculates and returns the average. The average is stored in a variable and printed in the main program.

**Daily Notes - Activity 2 - Write Methods**

 This has been completed and uploaded. It calculates a students average mark and determines if the student has passed, with a distinction, without a distinction or if they failed.

**My own views on lists and methods**

 Recursion is a method where the solution to a problem depends on the result of the same solution. In recursive functions, the function calls itself during execution.  
  
An example of a recursive function was demonstrated using the Fibonacci sequence. The Fibonacci sequence starts with 0 and 1, where each subsequent number is the sum of the previous two numbers. The Fibonacci function calculates the Fibonacci number for a given input using recursion.  
  
The program example showcased the Fibonacci function by printing the Fibonacci numbers for a range of values. The output demonstrated how the function recursively calculates the Fibonacci sequence.  
  
The Fibonacci recursive function effectively demonstrates the concept and power of recursion in solving problems that can be defined in terms of themselves.

**Daily Notes - Python Tutorial: File Objects - Reading and Writing to Files**

 Recursion is a method where the solution to a problem depends on the result of the same solution. In recursive functions, the function calls itself during execution.  
  
An example of a recursive function was demonstrated using the Fibonacci sequence. The Fibonacci sequence starts with 0 and 1, where each subsequent number is the sum of the previous two numbers. The Fibonacci function calculates the Fibonacci number for a given input using recursion.  
  
The program example showcased the Fibonacci function by printing the Fibonacci numbers for a range of values. The output demonstrated how the function recursively calculates the Fibonacci sequence.  
  
The Fibonacci recursive function effectively demonstrates the concept and power of recursion in solving problems that can be defined in terms of themselves.

**My Views on the Day**

 1. Learning about functions, the random function and the recursive function  
  
2. Activity 1, 2 & 3  
  
3. None  
  
4. None

**Daily Notes - Day 2 Reflections**

 1. Learning about functions, the random function and the recursive function  
  
2. Activity 1, 2 & 3  
  
3. None  
  
4. None