1. Variables and Comments (var.py)

```python
# Everything about Python
# Hash symbols are comments
# This is another comment
# The compiler / computer will not execute/run these lines

# 1. VARIABLES ================================================================
variable_name = 1 # variable_name is assigned the value 1 and is of type integer
anything = 'Hello' # anything is assigned the value 'Hello' and is of type string (alphabets)
number_variable = 1.0 # number_variable is assigned the value 1.0 and is of type float (decimal)
boolean_variable = True # boolean_variable is assigned the value True and is of type boolean (True or False)

# When you want to print the value of a variable, (OUTPUT)
# you can use the print function
print(variable_name)

# You can also print a string and a variable together
# You use commas to separate the string and the variable
print('The value of variable_name is:', variable_name)
print('The value of anything is:', anything)
print('The first variable is:', variable_name, 'and the second variable is:', anything)
```

2. INPUT (input.py)

```python
# 2. INPUT ====================================================================
# You can ask the user to input a value
# The input function will return a string
# You can assign the input to a variable
user_input = input('Enter a value: ') # The 'Enter a value' is a prompt
print('The user input is:', user_input)
# Notice the difference between having a prompt and not having a prompt
user_input1 = input()
print('The user input1 is:', user_input)
user_input2 = input('Your prompt: ')
print('The user input2 is:', user_input)
```

3. CONDITIONALS (condition.py)

```python
# 3. CONDITIONALS ============================================================
# You can use conditionals to check if a condition is true or false
# If the condition is true, the code inside the if block will be executed
# If the condition is false, the code inside the else block will be executed
# The condition is checked using the if keyword
# The else block is optional

# Structure of if - else
# if condition:
#     code
# else:
#     code

# Structure of if - elif - else
# if condition:
#     code
# elif condition:
#     code
# else:
#     code

# Just if condition. else is optional
if True: # Change to False and see if the code inside is executed or not.
    print('The condition is true')

variable_name = 1
# Basic if else
if variable_name == 1:
    print('The value of variable_name is 1')
else:
    print('The value of variable_name is not 1')

# if elif else
if variable_name == 1:
    print('The value of variable_name is 1')
elif variable_name == 2:
    print('The value of variable_name is 2')
else:
    print('The value of variable_name is not 1 or 2')
```

```python
# combining input with if else
user_name = input('What is your name: ')

print('Welcome ', user_name)
user_age = input('What is your age: ')
# The input() will return a string.
# You need to convert it to an integer using the int() function
user_age_int = int(user_age) # the string datatype is now converted to an integer datatype
# The above line is called typecasting; converting one datatype to another
# You can also convert it to a float using the float() function # In programming, we call decimal numbers as floats

# We typecasted because we want to compare numbers and not strings
if user_age_int < 18:
    print('You are a minor')
elif user_age_int >= 18 and user_age_int < 60: # Notice the 'and' keyword. It is used to combine two conditions
    print('You are an adult')
else:
    print('You are a senior citizen')

# condition1 and condition2: True only if both condition1 and condition2 are True
# condition1 or condition2: True if either condition1 or condition2 is True

# Combining multiple conditions (AND)
if True and True: # Only this will executed
    print('True True')

if True and False:
    print('True False')

if False and True:
    print('False True')

if False and False:
    print('False False')
```

4. LOOPS (loops.py)

```python
# 4. LOOPS ========================================================================
# Loops are used to repeat a block of code
# There are two types of loops: for and while
# for loop is used when you know the number of times you want to repeat the code
# while loop is used when you don't know the number of times you want to repeat the code

# 4.1 FOR LOOP ====================================================================

# Structure of for loop
# for element in iterable:
#     code

# iterable is a collection of items; Like array, tuples: collection of data

# for loop with a list
list_of_vegs = ['potato', 'tomato', 'onion', 'carrot', 'cabbage']
# Go through each element in the list and print it
for veg in list_of_vegs:
    print(veg)

# GO through each element. If element is potato, print it
for veg in list_of_vegs:
    if veg == 'potato':
        print(veg)

# for loop with a string
string = 'Hello World'
# Go through each character in the string and print it
for character in string:
    print(character)

# Go through each character in the string. If character is a vowen, print it
for character in string:
    if character == 'a' or character == 'e' or character == 'i' or character == 'o' or character == 'u':
        print('Vowel character', character)

# For loop with a range
# range(start, end)
# range(start, end, step)
# range(end)

# range(0, 10) will return a list of numbers from 0 to 9
for number in range(0, 10):
    print(number)
```

```python
# Print even numbers from 1 till 10
for number in range(1, 11):
    if (number / 2) == 0:
        print(number)

# Print odd numbers from 1 till 10
for number in range(0, 11):
    if (number / 2) != 0:
        print(number)

# Print numbers from 1 till 10 with a step of 2
for number in range(1, 11, 2):
    print(number)

# Print numbers from 10 till 1
for number in range(10, 0, -1):
    print(number)

# 4.2 WHILE LOOP ================================================================
# Structure of while loop
# while condition:
#     code

# while loop with a counter
counter = 0
while counter < 10:
    print(counter)
    counter = counter + 1 # counter += 1

# while loop with a list
list_of_vegs = ['potato', 'tomato', 'onion', 'carrot', 'cabbage']
counter = 0
# len(list_of_vegs) will return the length of the list;
print(len(list_of_vegs)) # This will print 5; Because there are 5 elements in the list
while counter < len(list_of_vegs):
    print(list_of_vegs[counter])
    counter = counter + 1

# while loop with a string
string = 'Hello World'
counter = 0
while counter < len(string):
    print(string[counter])
    counter = counter + 1
```

5. FUNCTIONS (fun.py)

```python
# 5. FUNCTIONS ================================================================
# Functions are used to group a set of statements together to perform a specific task
# Functions are used to avoid writing the same code again and again
# Functions are used to make the code more readable, reusable

# Structure of a function
# def function_name():
#     code

# Defining a function
def print_hello():
    print('Hello')

# Calling a function
print_hello()

# You can call a funtion inside a function
def print_hello_world():
    print_hello()
    print('World')

print_hello_world()

# Functions with parameters
# Parameters are used to pass data to a function

# Defining a function with parameters
def print_name(name):
    print('Hello', name)

# Calling a function with parameters
print_name('John')

# Functions with multiple parameters
def calculate_addition(x, y):
    print('Addition results: ', x + y)

calculate_addition(10, 20)

# Returning a value from a function
def calculate_addition(x, y):
    return x + y

result = calculate_addition(10, 20)
print('Addition results: ', result)

# Returning multiple values from a function
def calculate_addition_subtraction(x, y):
    return x + y, x - y

addition_result, subtraction_result = calculate_addition_subtraction(10, 20)
print('Addition results: ', addition_result)
print('Subtraction results: ', subtraction_result)
```

6. STRINGS (str.py)

```python
# 6. Manipulating Strings ============================================================
# Strings are immutable; You cannot change a string once it is created
# You can only create a new string from an existing string

# 6.1 CONCATENATION ==================================================================
# Concatenation is used to join two strings together
string1 = 'Hello'
string2 = 'World'
concatenated = string1 + string2
print(concatenated)

# 6.2 Operations on strings ==========================================================
# len(string): Returns the length of the string
string = 'Hello World'
str_length = len(string)
print(str_length)

# string[index]: Returns the character at the given index
# Index starts from 0
# Index can be negative
# Index can be out of range
string = 'Hello World'
print(string[0]) # H
print(string[2]) # l
print(string[5]) # ' '
print(string[10]) # d
print(string[-1]) # d
print(string[-4]) # o

# string[start:end]: Returns the characters from start index to end index
# end index is not included
string = 'Hello World'
print(string[0:5]) # Hello
print(string[6:11]) # World
print(string[0:11]) # Hello World
print(string[0:100]) # Hello World
print(string[0:-1]) # Hello Worl

# lower(): Converts the string to lower case
string = 'Hello World'
lower_cased = string.lower()
print(lower_cased)
# upper(): Converts the string to upper case
string = 'Hello World'
upper_cased = string.upper()
print(upper_cased)

# strip(): Removes the white spaces from the beginning and end of the string
string = '   Hello World   '
stripped = string.strip()
print(stripped)

# replace(old_string, new_string): Replaces the old string with the new string
string = 'Hello World'
replaced = string.replace('World', 'Universe')
print(replaced)
```

7. FILE I/O (file_io.py)

```python
# 7. FILE HANDLING ============================================================
# 7.1 Opening a file ==========================================================
# open(file_name, mode)
# file_name: Name of the file to be opened
# mode: Mode in which the file should be opened
# r: Read mode
# w: Write mode
# a: Append mode
# r+: Read and write mode
# w+: Write and read mode
# a+: Append and read mode

# Opening a file in read mode
# The variable 'file' is a File object with read access
file = open('file.txt', 'r')
# Print the contents of the file
contents = file.read() # Get me the contents of the file variable
print(contents)
file.close() # After opening a file, close to release it from the memory

# Writing to a file
file_variable_with_write_mode = open('new_file.txt', 'w')
file_variable_with_write_mode.write('This line will be written to the file')
file_variable_with_write_mode.close()

# THE EASIER WAY:
# reading a file
with open('file.txt', 'r') as file:
    file_text = file.readlines()

print('Easier way: ', file_text)

# writing to a file
with open('new_file.txt', 'w') as file:
    file.write('This line will be written to the file')

# Appending to a file; Adding to existing content; Not overwriting
# the mode is 'a' for append
with open('new_file.txt', 'a') as file:
    file.write('\nThis line will be appended to the file')
```