MOD 3 Day 1

Sunday, September 24, 2023 9:25 AM

Class Objectives

By the end of today's class, you will be able to:

- Perform Python 3 installation.
- Navigate through folders and files via the terminal.
- Create Python scripts and run them in the terminal.
- Understand basic programming concepts in Python.

Some basic commands

`cd`	changes the directory.
`cd ~`	changes to the home directory.
`cd ..`	moves up one directory.
`ls`	lists files in the folder.
`pwd`	shows the current directory.
`Mkdir <FOLDERNAME>`	creates a new directory with the FOLDERNAME
`touch <FILENAME>`	creates a new file with the FILENAME.
`rm <FILENAME>`	deletes a file.
`rm -r <FOLDERNAME>`	deletes a folder; note the -r.
`open .`	opens the current folder on Macs.
`explorer .`	opens the current folder on Windows.
`open <FILENAME>`	opens a specific file on Macs.
`explorer <FILENAME>`	opens a specific file on Windows.

1:40:00

He explains how to use command terminal.

Activty 2
8:07:00

Here we use terminal to create a file within terminal using code terminal code and we add a te
File with it and run it

02-Stu_TerminalTest > README.md > # Terminal

```
# Terminal

Now, it's your turn to do some work in the terminal. You'll create three folders and a pair of Python files to print strings of your own creation to the console.

## Instructions

Use the following instructions to write commands in your terminal:

* Create a folder called `LearnPython`.

* Navigate into the folder.

* Inside `LearnPython`, create another folder called `Assignment1`.

* Inside `Assignment1`, create a file called `quick_python.py`.

* Add a print statement to `quick_python.py`.

* Run `quick_python.py`.

* Return to the `LearnPython` folder.

* Inside `LearnPython`, create another folder called `Assignment2`.

* Inside `Assignment2`, create a file called `quick_python2.py`.

* Add a different print statement to `quick_python2.py`.

* Run `quick_python2.py`.

—

© 2022 edX Boot Camps LLC. Confidential and Proprietary. All Rights Reserved.
```

Activity 3
8:37:00
How to use variables
Use "f" to print your message

02-Stu_TerminalTest > Solved > $ terminal_games_solution.sh

```
# Follow the below instructions in your terminal and write the commands below.

# Create a folder called LearnPython
mkdir LearnPython

# Navigate into the folder
cd LearnPython
```

xt

```
# Navigate into the folder
cd LearnPython

# Inside LearnPython create another folder called Assignment1
mkdir Assignment1

# Inside Assignment1 create a file called quick_python.py
touch quick_python.py

# Add a print statement to quick_python.py
# add print("This file works!") in a python file

# Run quick_python.py
python quick_python.py

# Return to the LearnPython folder
cd ..

# Inside LearnPython create another folder called Assignment2
mkdir Assignment2

# Inside Assignment2 create a file called quick_python2.py
touch quick_python2.py

# Add a different print statement to quick_python2.py
# add print("This file also works!") in a python file

# Run quick_python2.py
python quick_python2.py
```

8:11:00
How to install anaconda

8:20:00
Virtual environment what are they and why we need them

8:22:00
How to create a virtual environment. Activate and deactivate it . (you can choose a diffent name that "dev")

Activity 4
Hello variable world
1:23:00

1:28:00
How to run a pyton .py file in terminlal

```
# Hello, Variable World

In this activity, you will create a simple Python application that uses variables to run calculations on integers and
print strings out to the console.

## Instructions

* Create two variables, called `name` and `country`, that will hold strings.

* Create two variables, called `age` and `hourly_wage`, that will hold integers.

* Create a variable called `satisfied`, which will hold a Boolean.

* Create a variable called `daily_wage`, which will hold the value of `hourly_wage` multiplied by 8.

* Use traditional string concatenation to print the `name`, `country`, `age`, and `hourly_wage` variables.

* With an `f-string`, print the `daily_wage` and `satisfied` variables.

## Hint

For additional help with f-strings, visit [Python 3's f-Strings](https://realpython.com/python-f-strings/).

-


```

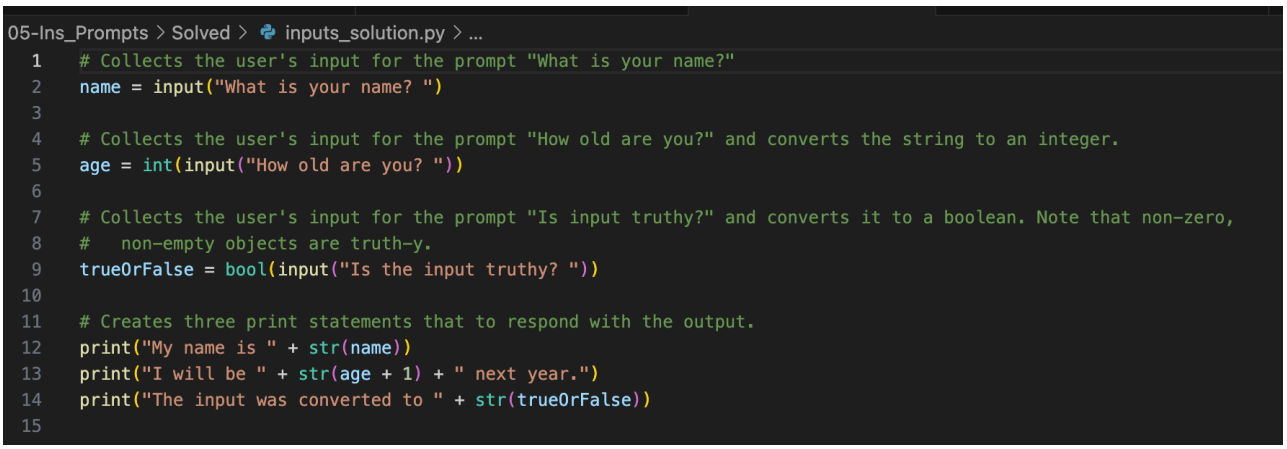
04-Stu_HelloVariableWorld > Solved > hello_variable_world_solution.py > ...

```python
# Create a variable called 'name' that holds a string
name = "Jacob Deming"

# Create a variable called 'country' that holds a string
country = "United States"

# Create a variable called 'age' that holds an integer
age = 25

# Create a variable called 'hourly_wage' that holds an integer
hourly_wage = 15

# Calculate the daily wage for the user
daily_wage = hourly_wage * 8

# Create a variable called 'satisfied' that holds a boolean
satisfied = True

# Print out "Hello <name>!"
print("Hello " + name + "!")

# Print out what country the user entered
print("You live in " + country)

# Print out the user's age
```

```
26    print("You are " + str(age) + " years old")
27
28    # With an f-string, print out the daily wage that was calculated
29    print(f"You make {daily_wage} per day")
30
31    # With an f-string, print out whether the users were satisfied
32    print(f"Are you satisfied with your current wage? {satisfied}")
33
```

Activuty 5
Promts
1:31:00

This takes a input from a prompt
Here we use bollon (bool) and int(age(num)) and we take the persons name
Then we output a series of statements.



Activity 6
Down to input
1:35:00

Here we take two inputs and take a another two follow up inputs then we display a min
And a calculation of both names.

The talk about how to use decimals and output a sring

06-Stu_DownToInput > README.md > # Down to Input

```
# Down to Input

In this activity, you store inputs from the command line and run code based on those inputs.

## Instructions

* Create two different variables, one to take the input of your first name and one for your neighbor's first name.

* Create two more inputs to ask how many months you and your neighbor have been coding.

* Finally, display a result with both your names and the total amount of months coding.

-


```

06-Stu_DownToInput > Solved > down_to_input_solution.py > ...

```python
# Take input of you and your neighbor
your_first_name = input("What is your name? ")
neighbor_first_name = input("What is your neighbor's name? ")

# Take how long each of you have been coding
months_you_coded = input("How many months have you been coding? ")
months_neighbor_coded = input("How many months has your neighbor been coding? ")

# Add total month
total_months_coded = int(months_you_coded) + int(months_neighbor_coded)

# Print results
print("I am " + your_first_name + " and my neighbor is " + neighbor_first_name)
print("Together we have been coding for " + str(total_months_coded) + " months!")

```

Activity 7
Conditionals

Is
If else
Nested if

In python you don't need to "end if" you can just move to the next
Indentation matter more than in VBA

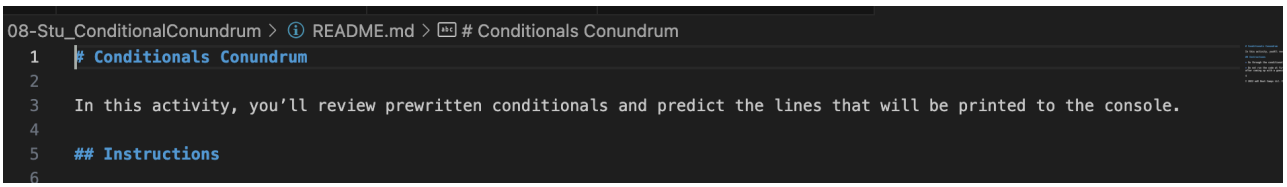
07-Ins_Conditionals > Solved > conditionals_solution.py > ...

```python
x = 1
y = 10

# Checks if one value is equal to another
if x == 1:
    print("x is equal to 1")

```

```python
# Checks if one value is NOT equal to another
if y != 1:
    print("y is not equal to 1")

# Checks if one value is less than another
if x < y:
    print("x is less than y")

# Checks if one value is greater than another
if y > x:
    print("y is greater than x")

# Checks if a value is greater than or equal to another
if x >= 1:
    print("x is greater than or equal to 1")

# Checks for two conditions to be met using "and"
if x == 1 and y == 10:
    print("Both values returned true")

# Checks if either of two conditions is met
if x < 45 or y < 5:
    print("One or more of the statements were true")

# Nested if statements
if x < 10:
    if y < 5:
        print("x is less than 10 and y is less than 5")
    elif y == 5:
        print("x is less than 10 and y is equal to 5")
    else:
        print("x is less than 10 and y is greater than 5")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

2:08:00
Activity 08
Conditional conundrum

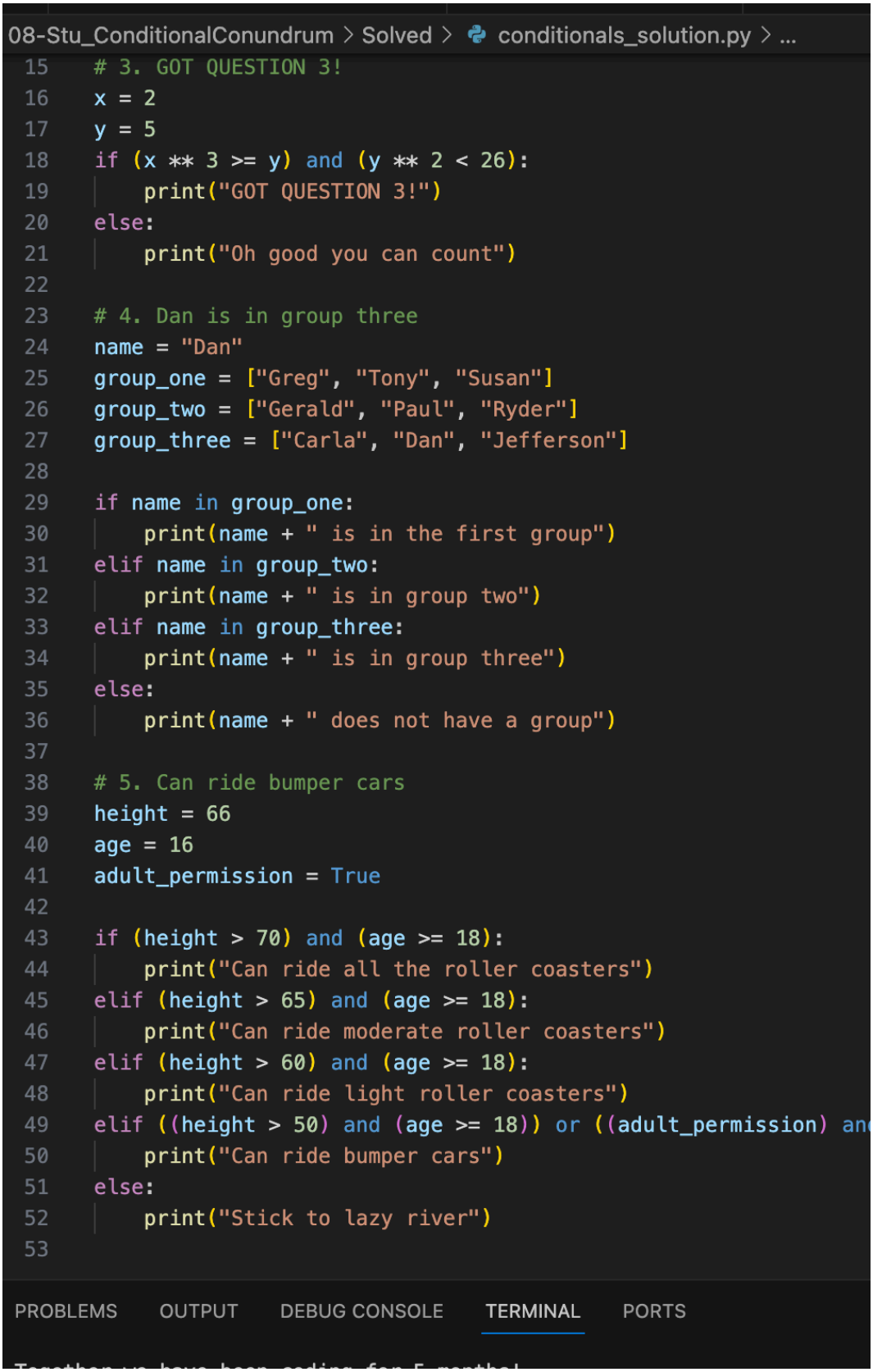


```
7   * Go through the conditionals in the provided code, and predict the lines that will be printed to the console.
8
9   * Do not run the code at first. Try to follow the thought process for each chunk of code and then make a guess. Only
    after coming up with a guess for each section should you run the application.
10
11  -
12
13  © 2022 edX Boot Camps LLC. Confidential and Proprietary. All Rights Reserved.
14
```

08-Stu_ConditionalConundrum > Solved > conditionals_solution.py > ...

```python
# 1. oooo needs some work
x = 5
if 2 * x > 10:
    print("Question 1 works!")
else:
    print("oooo needs some work")

# 2. Question 2 works!
x = 5
if len("Dog") < x:
    print("Question 2 works!")
else:
    print("Still missing out")

# 3. GOT QUESTION 3!
x = 2
y = 5
if (x ** 3 >= y) and (y ** 2 < 26):
    print("GOT QUESTION 3!")
else:
    print("Oh good you can count")

# 4. Dan is in group three
name = "Dan"
group_one = ["Greg", "Tony", "Susan"]
group_two = ["Gerald", "Paul", "Ryder"]
group_three = ["Carla", "Dan", "Jefferson"]

if name in group_one:
    print(name + " is in the first group")
elif name in group_two:
    print(name + " is in group two")
elif name in group_three:
    print(name + " is in group three")
else:
    print(name + " does not have a group")

# 5. Can ride bumper cars
height = 66
```

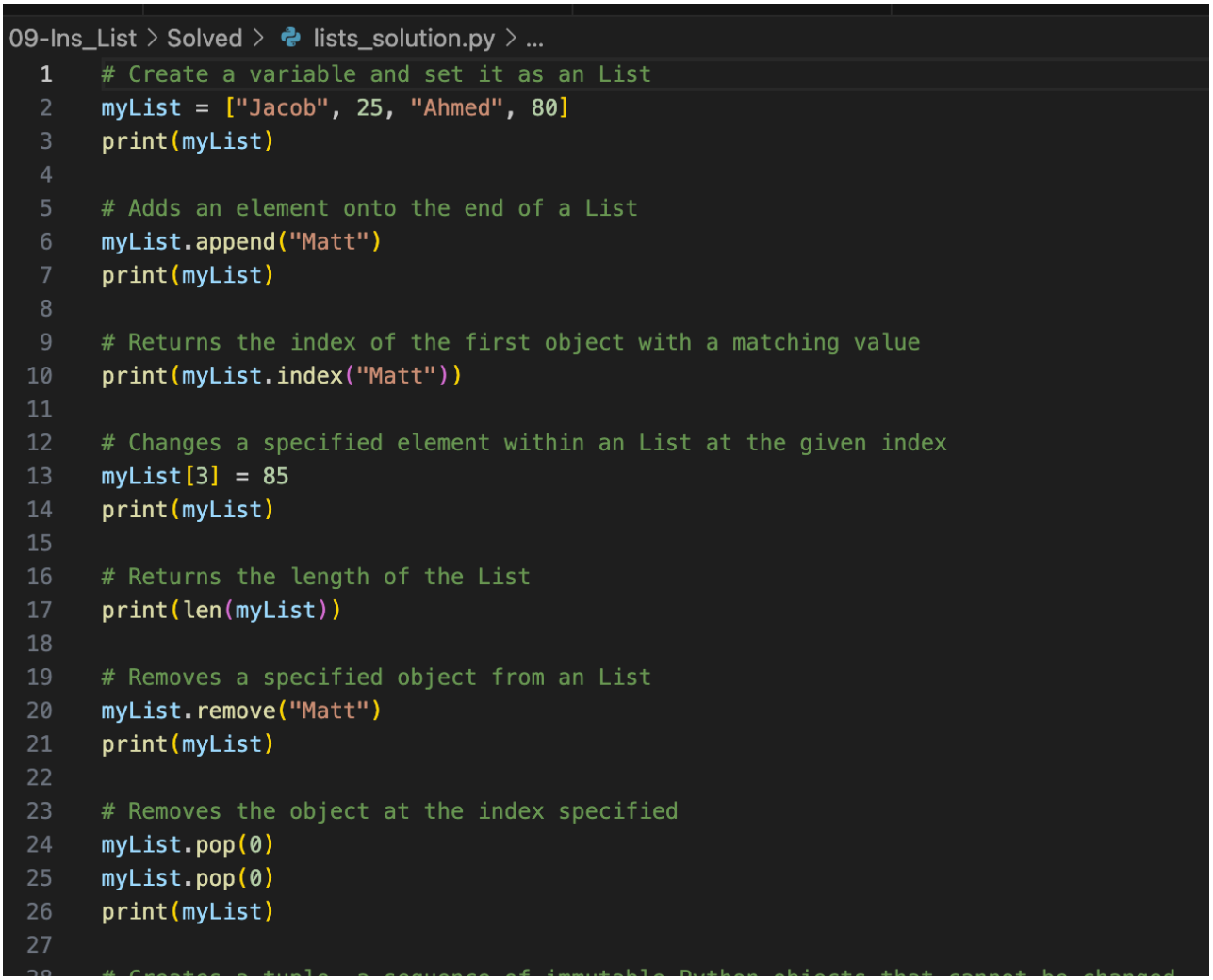
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS



PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Activity 09
2:25:00

The brackets are the diffence between tuples and list [list] and (tuples)
Tuples means no modification
Lists
We use square brackets to make lists
How to append a list
How to print an index of a list
How to remove from a list
How to remove object from list
Update a list
tuples and list are basically the same(only differnce is tuples cannopt be ammended)
tuples cannot be ammended removed and cannot be changed,,,, list can be channged



```python
# Creates a tuple, a sequence of immutable Python objects that cannot be changed
myTuple = ('Python', 100, 'VBA', False)
print(myTuple)

```

2:34:00
Activity 10
Rock paper scissor

This program uses 1 of 3 impute and outputs a random output
How to import library

10-Stu_RockPaperScissors > README.md > # Rock, Paper, Scissors

```
# Rock, Paper, Scissors

Create a Rock, Paper, Scissors game that takes user input from the command line and plays against the computer.

## Instructions

* Using the terminal, take an input of `r`, `p`, or `s` for rock, paper, or scissors.

* Have the computer randomly pick one of these three choices.

* Compare the user's input to the computer's choice to determine if the user won, lost, or tied.

## Hint

* Check out this [Stack Overflow](https://stackoverflow.com/questions/306400/how-to-randomly-select-an-item-from-a-list) question for help with using the `random` module to select a value from a list.

-

© 2022 edX Boot Camps LLC. Confidential and Proprietary. All Rights Reserved.

```

10-Stu_RockPaperScissors > Solved > rock_paper_scissors_solution.py > ...

```python
# Incorporate the random library
import random

# Print Title
print("Let's Play Rock Paper Scissors!")

# Specify the three options
options = ["r", "p", "s"]

# Computer Selection
computer_choice = random.choice(options)

```

```python
# User Selection
user_choice = input("Make your Choice: (r)ock, (p)aper, (s)cissors? ")

# Run Conditionals
if (user_choice == "r" and computer_choice == "p"):
    print("You chose rock. The computer chose paper.")
    print("Sorry. You lose.")

elif (user_choice == "r" and computer_choice == "s"):
    print("You chose rock. The computer chose scissors.")
    print("Yay! You won.")

elif (user_choice == "r" and computer_choice == "r"):
    print("You chose rock. The computer chose rock.")
    print("A smashing tie!")

elif (user_choice == "p" and computer_choice == "p"):
    print("You chose paper. The computer chose paper.")
    print("A smashing tie!")

elif (user_choice == "p" and computer_choice == "s"):
    print("You chose paper. The computer chose scissors.")
    print("Sorry. You lose.")

elif (user_choice == "p" and computer_choice == "r"):
    print("You chose paper. The computer chose rock.")
    print("Yay! You won.")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

10-Stu_RockPaperScissors > Solved > rock_paper_scissors_solution.py > ...

```python
    print("You chose rock. The computer chose paper.")
    print("Sorry. You lose.")

elif (user_choice == "r" and computer_choice == "s"):
    print("You chose rock. The computer chose scissors.")
    print("Yay! You won.")

elif (user_choice == "r" and computer_choice == "r"):
    print("You chose rock. The computer chose rock.")
```

```
(function) def print(
    *values: object,
    sep: str | None = " ",
    end: str | None = "\n",
    file: SupportsWrite[str] | None = None,
    flush: Literal[False] = False
```

```
33  elif ) -> None
34
35      print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)
36
37  elif Prints the values to a stream, or to sys.stdout by default. Optional keyword arguments:
38       file: a file-like object (stream); defaults to the current sys.stdout.
39       sep: string inserted between values, default a space.
40       end: string appended after the last value, default a newline.
         flush: whether to forcibly flush the stream.
41  elif (user_choice == "s" and computer_choice == "p"):
42      print("You chose scissors. The computer chose paper.")
43      print("Yay! You won.")
44
45  elif (user_choice == "s" and computer_choice == "s"):
46      print("You chose scissors. The computer chose scissors.")
47      print("A smashing tie!")
48
49  elif (user_choice == "s" and computer_choice == "r"):
50      print("You chose scissors. The computer chose rock.")
51      print("Sorry. You lose.")
52
53  else:
54      print("I don't understand that!")
55      print("Next time, choose from 'r', 'p', or 's'.")
56
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Activity 11
2:40:00

Loop though ranges list objects strings condition
While loop

11-Ins_Loops > Solved > loop_dee_loop_solution.py > ...

```
1   # Loop through a range of numbers (0 through 4)
2   for x in range(5):
3       print(x)
4
5   print("----------------------------------------")
6
7   # Loop through a range of numbers (2 through 6 - yes 6! Up to, but not inc
8   for x in range(2, 7):
9       print(x)
```

```
    print(x)

print("----------------------------------------")

# Iterate through letters in a string
word = "Peace"
for letter in word:
    print(letter)

print("----------------------------------------")

# Iterate through a list
zoo = ["cow", "dog", "bee", "zebra"]
for animal in zoo:
    print(animal)

print("----------------------------------------")

# Loop while a condition is being met
run = "y"

while run == "y":
    print("Hi!")
    run = input("To run again. Enter 'y': ")

```

Activity 12
Number chain
2:45:00

This program asks the user to choose a number and then prints the ascending number starting from 0…. When done listing, ask for another input and ask if you want to continue to count from that number

Bonus make the numbers start from the last chain

12-Stu_NumberChain-Loops > README.md > # Number Chain

```
# Number Chain

In this activity, you will take user input and print out a string of numbers.

## Instructions

* Using a `while` loop, ask the user "How many numbers?", and then print out a chain of numbers in increasing order,
from 0 to the user-input number.

* After the results have been printed, ask the user if they would like to continue
```

```
* After the results have been printed, ask the user if they would like to continue.

    * If "y" is entered, keep the chain running by inputting a new number and starting a new count from 0 to the new user-input number.

    * If "n" is entered, exit the application.

## Bonus

Rather than just displaying numbers starting from 0, have the numbers begin at the end of the previous chain.

-


```

12-Stu_NumberChain-Loops > Solved > number_chain_solution.py > ...

```python
# Initial variable to track game play
user_play = "y"

# While we are still playing...
while user_play == "y":

    # Ask the user how many numbers to loop through
    user_number = int(input("How many numbers? "))

    # Loop through the numbers. (Be sure to cast the string into an integer.)
    for x in range(user_number):

        # Print each number in the range
        print(x)

    # Once complete...
    user_play = input("Continue: (y)es or (n)o? ")

```

BONUS

12-Stu_NumberChain-Loops > Solved > number_chain_bonus_solution.py > ...

```python
# Initial variable to track game play
user_play = "y"

# Set start and last number
start_number = 0

# While we are still playing...
while user_play == "y":

    # Ask the user how many numbers to loop through
    user_number = input("How many numbers? ")

    # Loop through the numbers. (Be sure to cast the string into an integer.)
    for x in range(start_number, int(user_number) + start_number):

        # Print each number in the range
        print(x)

```

```
    # Set the next start number as the last number of the loop
    start_number = start_number + int(user_number)

    # Once complete...
    user_play = input("Continue the chain: (y)es or (n)o? ")

```