Python Assignment

# 1. The following line won't run because of a syntax error

# Fixed syntax error

print("helllo")

# 2. Exercise 2

# The following lines won't run properly,

# even if the syntax error in the line above is corrected,

# because of a run-time error

# Fixed runtime error

print("hell")

# 3. Display a string (greeting message) directly

print("Hello, welcome”)

# 4. Display the contents of a string variable

msg = "This is a string variable"

print(msg)

# 5. Display the string which contains single quotes

print("Indian's ")

# 6. Display the string which contains Double Quotes

print('Students, "Welcome ".')

# 7. Read two numbers and perform calculations

num1 = float(input("Enter the first number: "))

num2 = float(input("Enter the second number: "))

# Calculations

sum\_value = num1 + num2

difference = num1 - num2

product = num1 \* num2

quotient = num1 / num2

remainder = num1 % num2

power = num1 \*\* num2

print(f"Sum: {sum\_value}")

print(f"Difference: {difference}")

print(f"Product: {product}")

print(f"Quotient: {quotient}")

print(f"Remainder: {remainder}")

print(f"Power: {power}")

# 8. Check if num1 is an integer

if num1.is\_integer():

print("integer.")

else:

print("not an integer.")

# 9. Convert num1 to an integer

num1 = int(num1)

# 10. Find datatype for variables

print(type(num1))

print(type(num2))

# 11. Read a float value and print the number rounded to 2 decimal places

float\_value = float(input("Enter a float value: "))

print(f"Rounded value: {round(float\_value, 2)}")

# 12. Read a float value and print the absolute value

print(f"Absolute value: {abs(float\_value)}")

# 13. Store different types of values in variables

string\_val = "Hello"

numeric\_val = 42

complex\_val = 1 + 2j

list\_val= [1, 2, 3]

dict\_val = {"key": "value"}

set\_val = {1, 2, 3}

tuple\_val = (1, 2, 3)

# 14. Find the data type for the above variables

print(type(string\_val))

print(type(numeric\_val))

print(type(complex\_val))

print(type(list\_val))

print(type(dict\_val))

print(type(set\_val))

print(type(tuple\_val))

# 15. Display the number of letters in the string

greeting = "Welcome to Programming"

print(len(greeting))

# 16. Read first name and last name from the user and combine them

first\_name = input("Enter first name: ")

last\_name = input("Enter last name: ")

full\_name = first\_name + " " + last\_name

greeting\_message = "Hello, " + full\_name + "!"

print(greeting\_message)

# 17. Display the string with space

print(f"{first\_name} {last\_name}")

# 18. Display first two characters from the name

print(full\_name[:2])

# 19. Display last three characters from the name

print(full\_name[-3:])

# 20. Display 3rd character to last character

print(full\_name[2:])

# 21. Display 3rd to 5th character

print(full\_name[2:5])

# 22. Create a list of food with two elements

food = ["apple", "banana"]

# 23. Add one more to the food list

food.append("peach")

# 24. Add two more food strings

food.extend(["guava", "berry"])

# 25. Count total number of items in the list

print(len(food))

# 26. Print the first two items in food using slicing notation

print(food[:2])

# 27. Print the last item in food using index notation

print(food[-1])

# 28. Debug: Check if the number is odd or even

number = int(input("Enter a number: "))

if number % 2 == 0:

print("Even.")

else:

print(" Odd.")

# 29. Debug: Convert Centigrade to Fahrenheit

c = float(input("Enter temperature in Celcius: "))

f = 9 \* (c / 5) + 32

print("Temperature in Fahrenheit is:", f)

# 30. Debug: Calculate average of user inputs

count = int(input("Enter the count of numbers: "))

total\_sum = 0

for \_ in range(count):

x = int(input("Enter an integer: "))

total\_sum += x

avg = total\_sum / count

print("The average is:", avg)

# 31. Prove strings are immutable and lists are mutable

# Strings are immutable

str\_value = "Hello"

try:

str\_value[0] = 'h'

except TypeError as e:

print(f"Strings are immutable: {e}")

# Lists are mutable

list\_value = [1, 2, 3]

list\_value[0] = 100

print(f"Lists are mutable: {list\_value}")