Course: Introduction to Data Science (DS2006) - Laboratory 04

Task 1-2: battle-of-dices-bad-not-rigged.py submitted

Task 3-5: battle-of-dices-not-rigged.py submitted

Task 6:

```
x = 5
y = 3
x = y
print(x)

# Output
# 3

```

3 is logged because x is assigned to the value of y which is 3.

Task 7:

```
x = 5
y = 3
y = x
print(y)

# Output
# 5

```

The output is the result of y being re-assigned to the value of x which is 5.

Task 8:

```
x = 10
y = x
x = 20
print(y)

# Output
# 10

```

Since y was not reassigned to x after changing the value of x, y is still referencing the value of x before x was changed to 20. This results in 10 being outputted instead of 20

Task 9:

```
x = 10
y = x
x = 20
y = x
print(y)

# Output
# 20

```

Assigning y to x means the value of x (the latest value assigned) is outputted, in this case 20.

Task 10:

```
1   x = "5"
2   y = 3
3   print (x * y)
4
5   # Output
6   # 555
7
```

The reason the output is 555 is because the string 5 is being written 3 times instead of being multiplied by 5 as it would if x was an integer. I was expecting a runtime error due to x and y being different data types, so it surprised me to see that it wrote 5 three times instead.

Task 11:

```
1   x = 7
2   y = 2
3   print("x / y")
4
5   # Output
6   # x / y
7
```

As shown in the image above the print function logs the string “x / y” as an independent value rather than the product of x and y. This is expected because the print function takes arguments, and passing a string leads to that string being logged instead of the division of x and y.

Task 12:

```
1   x = 7
2   y = 2
3   print(x / y)
4
5   # Output
6   # 3.5
7
```

This output shows the intended result from task 11 but written correctly. It divides and prints the product of x and y. Like task 11 this output is also expected because the argument to the print function is a calculation and thus the result is what is printed onto the console.

Task 13:

```
1   x = 12
2   y = 5
3   print(x // y)
4
5   # Output
6   # 2
7
```

The double slash shown in the above code snippet is a floor division, which rounds down the product to the nearest whole number if the product is a positive number, while for negative numbers it rounds to the nearest whole negative number towards negative infinity meaning if the product is -3.5 it rounds to -4.

Task 14:

```
x = 7
y = 2
print(x % y)

# Output
# 1

```

In the code snippet shown above, the “%” symbol is a modulo operator. It returns the remainder of the division. In the example shown above it returns the remainder of 7 / 2 which is 1.

Task 15:

```
x = 2
y = "3"
print(str(x) + y)

# Output
# 23

```

```
x = "2"
y = "3"
print(x + y)

# Output
# 23

```

The outputs of figure 12 and 13 are the same because both x and y are strings and are thus concatenated.

Task 16:

```
x = 10
if x > 5:
    print("Big")
else:
    print("Small")

# Output
# Big

```

Since x is 10 and 10 is greater than 5, “Big” is logged on the console.

Task 17:

```
x = 100
if x > 5:
    print("Big")
else:
    print("Small")

# Output
# Big

```

“Big” is also logged in this case because x is still greater than 5.

Task 18:

```
x = -10
if x > 5:
    print("Big")
else:
    print("Small")

# Output
# Small

```

Now that x is smaller than 5, “Small” is outputted.

Task 19 & 20:

```
x = 3
if x == 5:
    print("Five")
elif x <= 5:
    print("Less")
else:
    print("More")

# Output
# Less

```

The elif condition is outputted this time because 3 is less than 5 which meets the condition of x <= 5.

Task 21:

```
x = 5
if x == 5:
    print("Five")
elif x <= 5:
    print("Less")
else:
    print("More")

# Output
# Five

```

This time the first condition is met and thus outputted because x, which is 5, is equal to 5.

Task 22:

```
x = 10
y = 2
if x > 20:
    print("A")
elif y > 5:
    print("B")
elif x > 0:
    print("C")
else:
    print("D")

# Output
# C

```

Because x is less than 20 the first condition is false and so the second condition is tried. y is smaller than 5 which means the next condition is checked. The third condition is met as 10 is greater than 0 and thus the output becomes “C”.

Task 23:

```
x = 10
y = 20
if x > 20:
    print("A")
elif y > 5:
    print("B")
elif x > 0:
    print("C")
else:
    print("D")

# Output
# B

```

Following the same logic as task 22, the output becomes “B” due to y(20) being greater than 5.

Task 24:

```
x = -5
y = -3
if x > 20:
    print("A")
elif y > 5:
    print("B")
elif x > 0:
    print("C")
else:
    print("D")

# Output
# D

```

“D” is outputted due to all 3 conditions not being met which results in the else block to trigger.

Task 25:

```
names = ["Laura", "Yasmin"]
for each_name in names:
    print(each_name)

# Output
# Laura
# Yasmin

```

Each name in the names list is outputted on their respective rows.

Task 26:

```
names = ["Laura", "Yasmin"]

newName = input("Enter a new name: ")
names.append(newName)

for each_name in names:
    print(each_name)

# Output
# Laura
# Yasmin
# newName (e.g. John Doe)

```

The new name is added to the end of the list and is outputted last after the existing names.