I. Оператори порівняння та логічні оператори. Умови.

1. Створіть цілочисленну змінну зі своїм щасливим або улюбленим числом. Напишіть код, який би автоматизовував наступну задачу.

Якщо число парне, помножте його на **2** та виведіть результат на екран.

Але якщо число непарне, помножте його на **3** та додайте **1** і виведіть на екран.

In [9]:

```
luckynum = 7

if(luckynum % 2 == 0):
    print(luckynum * 2)
else:
    print(luckynum * 3 + 1)
```

```
22
```

1. Створіть змінну, що посилається на список. Якщо список пустий, додайте будь-який елемент у кінець списку. Якщо ж ні, то нічого не робіть.

In [14]:

```
my_list = []
if my_list.__len__() == 0:
    my_list.append(False)

print(my_list)
```

```
[False]
```

1. Створіть рядок зі своїм ім'ям латинкою. Напишіть код, який би автоматизовував наступну задачу.

Якщо в імені є літера **"a"** або **"i"**, Виведіть на екран повідомлення **"A or I is in the name"**.

Якщо в імені є літери **"o"** та **"n"**, Виведіть на екран повідомлення **"O and N is in the name"**.

Також якщо в імені немає літери **"o"**, але є **"n"**, Виведіть на екран повідомлення **"Only N is in the name"**.

В решті випадків виведіть на екран повідомлення **"There is no N in the name, but there might be O"**.

Зверніть увагу, що, наприклад, для імені **"Antonina"** буде виведено два повідомлення.

In [1]:

```
name = "Artemii"

name = name.lower()
if 'a' in name or 'i' in name:
    print("A or I is in the name")
if 'o' in name and 'n' in name:
    print("O and N is in the name")
elif 'n' in name:
    print("Only N is in the name")
else:
    print("There is no N in the name, but there might be O")
```

```
A or I is in the name
There is no N in the name, but there might be O
```

II. Comprehensions.

1. Створіть список, використовуючи **if** (не **if-else**), **range** та **comprehension.**

In [15]:

```
compList = [i for i in range(10) if i % 2 == 0]
print(compList)
```

```
[0, 2, 4, 6, 8]
```

1. Створіть словник, використовуючи **dict comprehension,** що містить **if else.** Коли слід використовувати **comprehensions,** а коли їх слід уникати**?**

In [19]:

```
compDict = {i**2: True if i % 2 == 0 else False for i in range(10)}
print(compDict)
```

```
{0: True, 1: False, 4: True, 9: False, 16: True, 25: False, 36: True, 49: False, 64: True
, 81: False}
```

Comprehensions варто використовувати для простих, коротких, "очевидних" операцій, об'єктивно доречних застосувань небагато: просте фільтрування та/або перетворення. Їх слід уникати якщо є велика кількість та/або складні умови; якщо паралельно потрібно виконати якусь дію відповідно до умови або ввести додаткові змінні, **comprehension** "не вистачить".

1. Створіть список з рядками та запишіть його у змінну. Створіть ще один список на основі попереднього, де кожен рядок буде містити тільки три перших заглавних символи. Наприклад: **['hello', 'i', 'dont', 'care'] -> ['HEL', 'I', 'DON', 'CAR'].** Використання **comprehensions** обов'язкове.

In [21]:

```
strings = "Створіть список з рядками та запишіть його у змінну. Створіть ще один список н
а основі попереднього, де кожен рядок буде містити тільки три перших заглавних символи.".
split()
strs = [i[:3].upper() for i in strings]
print(strs)
```

```
['СТВ', 'СПИ', 'З', 'РЯД', 'ТА', 'ЗАП', 'ЙОГ', 'У', 'ЗМІ', 'СТВ', 'ЩЕ', 'ОДИ', 'СПИ', 'НА
', 'ОСН', 'ПОП', 'ДЕ', 'КОЖ', 'РЯД', 'БУД', 'МІС', 'ТІЛ', 'ТРИ', 'ПЕР', 'ЗАГ', 'СИМ']
```

III. Цикли.

1. Продемонструйте роботу циклу **while.** Не можна використовувати **True** для запуску циклу, щоб уникнути безкінечного циклу**!**

In [39]:

```
import random

j = 0.1
while j < 10:
    i = random.random() + 1
    j *= i
    print((int(j) + 1)*"*")
```

```
*
*
*
*
*
**
**
```

```
***
****
*******
*********
***************
```

1. Напишіть програму, яка виведе на екран непарні числа в діапазоні від **0** до **20** включно.

In [40]:

```
print([i for i in range(0, 21) if i % 2 == 1])
```

```
[1, 3, 5, 7, 9, 11, 13, 15, 17, 19]
```

1. Створіть список з елементами булевого типу або **None.** Використовуючи цикли, отримайте в результаті список з кортежами, де перший елемент кортежу - індекс(ціле число), а другий елемент - відповідне значення з першого списку.

Наприклад, **[True, True, None, False] -> [(0, True), (1, True), (2, None), (3, False)]**

In [41]:

```
boolList = [True, True, None, True, False, None, False, False, True, None]
tupleList = []

j = 0
for i in boolList:
    tupleList.append((j, i))
    j += 1
print(tupleList)
```

```
[(0, True), (1, True), (2, None), (3, True), (4, False), (5, None), (6, False), (7, False
), (8, True), (9, None)]
```

1. Створіть словник, де ключі - назви книжок, а значення - їхня кількість у наявності в Вашій міні-бібліотеці. Бібліотека має містити щонайменше **6** книжок **(6** пар значень у словнику**)** і щонайбільше **10 (**пар значень**).** Використовуючи цикли, оновіть словник **(**не створюйте новий**)** так, щоби кількість книг у наявності збільшилося на **5** кожної книги. Наприклад, **{'It': 3, 'Fault stars': 10, 'Bible': 17, 'Psychological romance': 4, 'Harry Potter': 13} -> {'It': 8, 'Fault stars': 15, 'Bible': 22, 'Psychological romance': 9, 'Harry Potter': 18}**

In [43]:

```
lib = {"HGTTG": 23, "1984": 12, "Castle": 5, "The Double": 7, "Princess and the frog": 3
, "Cinderella": 1}

for book in lib:
    lib[book] += 5

print(lib)
```

```
{'HGTTG': 28, '1984': 17, 'Castle': 10, 'The Double': 12, 'Princess and the frog': 8, 'Ci
nderella': 6}
```

1. Визначте цілочислену змінну **n,** що належить проміжку від **4** до **10** включно. Використовуючи цикли, виведіть в консоль наступний патерн.

#

# #

# #

# #

... #*n

In [2]:

```
n = 6

for i in range(1, n + 1):
    print(i*"#")
```

```
#
##
###
####
#####
######
```

1. Гра **"Нумо вгадай".** Розробіть просту гру**,** де користувач має вгадати випадкове число від **1** до **100.** Для генерації випадкового числа використовуйте функцію **random.randint(a, b) (**тут **a** та **b** включно**).** Для отримання числа з консолі використовуйте функцію **input(),** результат якої обов'язково явно приведіть до типу **int.**

Підказка**:** використовуйте цикл **while** та умови **if-else,** щоб повідомити користувачу піказки **(**напр**.** "Більше"**,** "Менше"**)** поки користувач не вгадає**.** По завершенню виведіть кількість спроб**,** які знадобилися для вгадування**.**

In [54]:

```
NUMBER = random.randint(1, 100)
guess = -1
guesses = 1
print("Вгадайте число від 1 до 100")
while guess != NUMBER:
    guess = int(input())
    if guess > NUMBER:
        print("Менше")
        guesses += 1
    elif guess < NUMBER:
        print("Більше")
        guesses += 1
    else:
        print(f"Кількість спроб: {guesses}")
```

Вітаю! Ви велика(ий) молодець, що впоралась(вся). Похваліть себе та побалуйте чимось приємним. Я Вами пишаюся.