3. ЦИКЛЫ

3.1. Расширенные операторы присваивания

Если переменную необходимо увеличить (или уменьшить) на (или в) какое-то число раз, Python позволяет просто заменить предыдущее значение переменной на новое. Например,

```
counter = counter + 1
summa = summa % 2
```

Чтобы не приходилось писать имя переменной дважды, используют расширенные операторы присваивания. Для примеров выше:

```
counter += 1
summa %= 2
```

Примечание. При записи расширенного оператора между знаком арифметической операции и знаком равенства пробела нет. Иначе будет выведено сообщение об ошибке *SyntaxError: invalid syntax.*

Оператор	**Пример использования**	**Эквивалент**
+=	x += 5	x = x + 5
-=	x -= 2	x = x - 2
*=	x *= 10	x = x * 10
/=	x /= 4	x = x / 4
//=	x //= 4	x = x // 4
%=	x %= 4	x = x % 4
**=	x**=3	x=x**3

```
'''
Пример 3.1.1) В ходе эксперимента два параметра изменяются следующим образом: вначале первый увеличивается на 5, затем к результату прибавляется второй параметр, после чего полученное значение умножается на второй и, наконец, вычисленный результат нацело делится на второй параметр.
'''
a, b = int(input('a: ')), int(input('b: '))
print('Увеличиваем a на 5: a += 5,')
a += 5
print('a =', a)
print('Увеличиваем новое значение a на b: a += b')
a += b
print('теперь a =', a)
print('Увеличиваем новое значение a в b раз: a *= b,')
a *= b
print('получаем a =', a)
print('И находим целую часть от деления a на b: a //= b,')
a //= b
print('окончательно, a =', a)
```

(Пр)Задача 3.1.1) Присвоить переменной k значение 3,5. Используя расширенные операторы присваивания, последовательно изменить значение переменной: а) увеличить на 6,5; б) уменьшить результат в 2 раза; в) возвести результат в 7 степень. Результат вывести с 5 знаками после запятой.

<u>Задачи для самостоятельного решения.</u>

Задача 3.1. Переменной *n* присвоить номер Вашего варианта. Используя расширенные операторы присваивания, последовательно изменять значение переменной (не используя вспомогательные переменные):
а) увеличить в 7 раз;
б) разделить нацело на 5;
в) возвести результат в (-5) степень
г) найти остаток от деления результата на номер Вашего варианта.

Задача 3.2.
Написать программу, которая присваивает переменной год Вашего рождения, затем результат делит нацело на день рождения, а к итогу прибавляет удвоенный месяц рождения. Использовать ровно одну переменную и расширенный оператор присваивания.

3.2. Цикл for

Цикл в Python задает повторяющиеся действия. Возможны две ситуации: когда требуется повторить набор операторов заранее заданное число раз (применяется цикл *for*), и случаи, когда число повторов неизвестно, но задано некоторое условие, при котором воспроизведение действий прекращается (цикл *while*).

Рассмотрим вначале цикл *for*.

Формат записи:

```
for i in range(количество_повторений):
    тело_цикла
```

Примечание 1. Здесь *i* – переменная цикла. Может иметь, вообще говоря, любое доступное для переменной имя.

Примечание 2. Однократное выполнение тела цикла называется итерацией цикла*.*

Примечание 3. Как и для записи условного оператора, при описании цикла *for* обязательно ставить двоеточие в конце первой строки,
а все операторы, включенные в тело цикла, начинать с отступа в четыре пробела.

```
# Пример 3.2.1) Рассмотрим код, который распечатает
# 5 раз высказывание Ф. Бэкона "Знание - сила":
for i in range(5):
    print('Знание - сила')
```

Цикл *for* позволяет упростить ввод данных с клавиатуры и их однотипную обработку.

```
'''
Пример 3.2.2) Программа считывает данные 3-х испытаний
и возводит каждое в квадрат.
'''
for i in range(3):
    num = float(input())
    print(num, '^ 2 = ', num ** 2) # повторяются в теле
цикла
print('Выполнено') # команда написана без отступа,
поэтому значение выводится 1 раз, после окончания
цикла.
```

```
'''
Пример 3.2.3) Программа считывает с клавиатуры
результаты 10 испытаний и находит среди них наименьшее.
'''
min_res = float(input())
for i in range(10):
    res = float(input())
    if res < min_res:
        min_res = res
print('минимальный результат: ', min_res)
```

(Пр)Задача 3.2.1) Написать программу, которая 15 раз «повторит» (выведет на экран) фразу «Кеша хор-роший». Каждый раз с новой строки.

(Пр)Задача 3.2.2) Написать программу, которая запрашивает с клавиатуры целое число и выводит его такое число раз, чему оно равно, если оно четное и вдвое большее число раз, если – нечетное.

Задачи для самостоятельного решения.

Задача 3.3. Написать программу, которая помогает студенту подготовиться к занятию и для лучшего закрепления знаний «повторяет» 100 раз указанный термин (выводит каждый раз с новой строки):

В-1. Адаптер - это устройство связи компьютера с периферийными устройствами.

В-2. Алгоритм - это описание последовательности действий, строгое исполнение которых приводит к решению поставленной задачи за конечное число шагов.

В-3. База данных - один или несколько файлов данных, предназначенных для хранения, изменения и обработки больших объемов взаимосвязанной информации.

В-4. Бит - это самая маленькая единица измерения информации. 1 Байт=8 Бит. Для хранения 1 символа на компьютере необходимо 8 Бит, т. е. 1 Байт.

В-5. Браузер - (англ. browse - листать, просматривать) это программа, служащая для просмотра web-документов и обеспечивающая переход с одной web-страницы на другую в соответствии с гиперссылкой.

В-6. Видеокарта - электронная плата, которая обрабатывает видеоданные (текст и графику) и управляет работой дисплея.

В-7. Вирус - это размножающаяся программа, написанная человеком для нанесения ущерба ПК.

В-8. Гиперссылка - это выделенный объект, связанный с другим файлом и реагирующий на щелчок мыши.

В-9. Домен - это некий логический уровень Интернета, то есть группа сетевых ресурсов, имеющая собственное имя и управляемая своей сетевой станцией.

В-10. Драйвер - это программа сопряжения устройства с компьютером.

В-11. Запрос - это задание пользователем вопроса о нужной ему информации в той системе, в которой осуществляется работа (на сайтах или поисковых системах, в базах данных).

В-12. Информационная система - это взаимосвязанная совокупность средств, методов и персонала, используемая для сохранения, обработки и выдачи информации с целью решения конкретной задачи.

В-13. Информационная культура - умение работать с информацией и использовать для ее получения, обработки и передачи современные технические средства и методы.

В-14. Кластер - это группа секторов для размещения на диске больших объемов связанной информации, рассматриваемый операционной системой как единая область.

В-15. Каталог - объект в файловой системе, упрощающий организацию файлов; позволяет сгруппировать файлы и, возможно, другие каталоги.

В-16. Команда - это описание операции, которую должен выполнить компьютер.

В-17. Компьютер - (англ. computer — вычислитель) это программируемое электронное устройство, способное обрабатывать данные и производить вычисления, а также выполнять другие задачи манипулирования символами.

В-18. Монитор - это устройство вывода информации.

В-19. Мультимедиа - это сочетание звука, музыки, текста, графики и видео с возможностью управления.

В-20. Программирование - это процесс составления компьютерной программы на основе определенного алгоритма.

Задача 3.4. Если введено число, большее номера Вашего варианта, то вывести на экран номер варианта столько раз, каков Ваш год рождения, иначе, - 3 раза повторить день Вашего рождения. Выводить числа в одну строку, через символ «*».

Задача 3.5. При проведении исследования результаты испытания закодировали буквами A, B, C, D и E. Написать программу, которая печатает в столбец код результата. Для вывода повторяющихся значений использовать цикл.

B-1.	B-2.	B-3.	B-4.	B-5.	B-6.	B-7.	B-8.	B-9.	B-10.
A	A	A	A	A	A	A	A	A	A
B	B	B	A	B	B	A	B	A	A
C	C	B	A	B	C	A	B	A	B
C	D	B	A	B	C	A	B	A	B
C	D	C	A	C	C	B	B	A	B
C	D	D	B	D	C	C	B	A	B
D	D	E	C	D	C	D	C	A	B
D	D	E	C	D	C	E	C	B	B
E	D	E	C	D	C	E	C	C	B
	D	E	C	D	D	E	C	C	B
	D	E	D	D	E	E	D	C	C
	D	E	E	E	E	E	E	C	D
	E	E	E	E	E	E	E	D	E
	E	E	E	E	E	E	E	E	
	E	E	E	E	E	E	E		
	E	E							

B-11.	B-12.	B-13.	B-14.	B-15.	B-16.	B-17.	B-18.	B-19.	B-20.
A	A	A	A	A	A	A	A	A	A
B	B	B	A	A	A	B	B	B	B
B	B	C	A	B	A	C	C	C	C
B	B	C	A	B	A	D	C	D	C
C	C	C	A	B	A	D	C	D	D
D	C	C	A	B	B	E	C	D	E
D	C	D	A	B	C	E	C	D	E
D	C	E	A	B	D	E	C	D	E
E	C	E	A	B	D	E	C	D	E
	C	E	B	B	D	E	C	D	E
	D	E	C	B	D	E	D	D	E
	E		D	B	E		D	D	E
			D	B			D	D	E
			D	C			D	D	E
			D	D			E	D	E
			D					E	E
			E					E	

3.3. Изменение переменной цикла

Для задания переменных цикла традиционно используют буквы *i*, *j*, *k* или нижнее подчеркивание «_», если обращение к переменной в теле цикла не предполагается.

Рассмотрим на примере, как изменяется переменная при выполнении цикла.

```
# Пример 3.3.1) Вывод на экран переменной цикла
for i in range(5):
    print(i)
```

Примечание. Переменная цикла «пробегает» все значения из заданного диапазона. В примере выше последовательность всех целых чисел **от 0 до *n*-1** генерируется при помощи функции *range*(*n*).

```
# Пример 3.3.2) Вывод нумерованного списка
пользователей.
for i in range(10):
    x = input('Введите ФИО:')
    print('Пользователь №', i, x)
```

(Пр)Задача 3.3.1) Написать программу, которая выводит на экран все последовательные целые числа, кратные 5, начиная с 0, до числа, совпадающего с текущим годом, в одну строку через пробел.

<u>Задачи для самостоятельного решения.</u>

Задача 3.6. Написать программу, которая выводит на экран кубы всех целых чисел, начиная с 0, до числа вашего рождения включительно в формате:

0 -> 0
1 -> 1
2 -> 8
3 -> 27
…

Задача 3.7. Напишите программу, которая выводит таблицу умножения всех цифр на номер вашего варианта. Результат выведите в формате:

0 x [номер варианта] = 0
1 x [номер варианта] = …

3.4. Полная структура оператора for

Если необходимо, чтобы переменная цикла изменялась в иных пределах, чем описанные выше, то применяется полная форма записи функции *range* с тремя параметрами.

Формат записи:

```
for i in range(start, stop, step):
    тело_цикла
```

где *start* – начальное значение (по-умолчанию равен 0);

stop – конечное значение (всегда не включено);

step – шаг (по-умолчанию равен 1).

Примечание.

-функция *range*() с одним аргументом, означает, что задан параметр *stop*.

-функция *range*() с двумя аргументами, означает, что заданы *start* и *stop*.

-функция *range*() с тремя аргументами, означает, что заданы *start, stop, step*.

Примечание 1. Если первый параметр больше второго, то функция *range*() генерирует пустую последовательность.

Примечание 2. step может быть отрицательным, но тогда *start* должен быть больше, чем *stop*. Пустая последовательность в этом случае генерируется, если первый параметр меньше второго.

Примечание 3. Функция *range*() может генерировать только целые числа, включая отрицательные.

Примечание 4. Величина шага не может равняться нулю – это приведет к ошибке *ValueError*.

Примеры использования функции range()

Вызов функции	Последовательность чисел
`range(10)`	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
`range(1, 10)`	1, 2, 3, 4, 5, 6, 7, 8, 9
`range(3, 7)`	3, 4, 5, 6
`range(7, 3)`	пустая последовательность
`range(2, 15, 3)`	2, 5, 8, 11, 14
`range(9, 2, -1)`	9, 8, 7, 6, 5, 4, 3

Вызов функции	Последовательность чисел
range(3, 10, -2)	пустая последовательность

```
# Пример 3.4.1) Примеры работы цикла for.
print('range(1,11)')
print('i: ',end=' ')
for i in range(1,11):
    print(i,end=' ')

print(end='\n\n')
print('range(-5,55)')
print('i: ',end=' ')
for i in range(-5,55):
    print(i,end=' ')

print(end='\n\n')
print('range(11,1)')
print('i: ',end=' ')
for i in range(11,1):
    print(i,end=' ')
print(end='-пустая последовательность')

print(end='\n\n')
print('range(1,11,2)')
print('i: ',end=' ')
for i in range(1,11,2):
    print(i,end=' ')

print(end='\n\n')
print('range(-5,55,10)')
print('i: ',end=' ')
for i in range(-5,55,10):
     print(i,end=' ')

print(end='\n\n')
print('range(11,1,-1)')
print('i: ',end=' ')
for i in range(11,1,-1):
    print(i,end=' ')

print(end='\n\n')
print('range(55,-5,-10)')
print('i: ',end=' ')
```

```
for i in range(55,-5,-10):
    print(i,end=' ')



print(end='\n\n')
print('range(1,11,-1)')
print('i: ',end=' ')
for i in range(1,11,-1):
    print(i,end=' ')
print(end='-пустая последовательность')
```

(Пр)Задача 3.4.1) Используя циклы, напишите программу, которая выведет последовательность:

-4 -2 0 2 4 6 8 10 7 6 5 4 3

<u>Задачи для самостоятельного решения.</u>

Задача 3.8. Написать программу, которая выводит числа из промежутка от дня Вашего рождения до года Вашего рождения включительно, с шагом, равным Вашему варианту, в одну строку, через символ «;».

Задача 3.9. Написать программу, которая выводит в порядке убывания все целые числа из промежутка (используйте отрицательный шаг цикла):

В-1. [–10; 10]
В-2. (–5; 4]
В-3. [–8; 7)
В-4. (–9; 11)
В-5. [–3; 18]
В-6. (–6;15]
В-7. [–11;5)
В-8. (–4; 12)
В-9. [–7; 14]
В-10. (–11; 3]
В-11. [–9; 10)
В-12. (–13; 16)
В-13. [–17; 5]
В-14. (–14; 4]

B-15. [–7; 25)
B-16. (–23; 8)
B-17. [–12; 22]
B-18. (–34; 3]
B-19. [–11; 55)
B-20. (–44; 4)

3.5. Цикл while

Цикл *while* нужен тогда, когда заранее не известно число итераций.
Формат записи:

```
while условие:
    тело_цикла
```

(пока условие, стоящее до обязательного двоеточия, выполняется, – делай команды, записанные с отступом в четыре пробела, как только условие выполнено – выход из цикла).

Примечание 1. Если условие ложно до цикла, то не будет произведено ни одной итерации.

Примечание 2. Если условие не может быть выполнено, получится бесконечный цикл. В теле цикла должно быть предусмотрено изменение условия на ложное, некоторое стоп-значение.

Примечание 3. Условие может содержать логические операции *and, or, not.*

```
'''
Пример 3.5.1) Программа распечатает 5 раз высказывание
Ф. Бэкона "Знание - сила" при помощи цикла while
(аналог for):
'''
i = 0
while i < 5:
    print('Знание - сила')
    i += 1
```

(Пр)Задача 3.5.1) Написать программу, которая нумерует и выводит на экран все введенные с клавиатуры строки, до тех пор, пока не введено «@&@» (без внешних кавычек).

(Пр)Задача 3.5.2) С клавиатуры вводятся действительные числа. Написать программу, которая выводит на экран их квадраты с одним знаком после запятой, пока не введен 0.

Задачи для самостоятельного решения.

Задача 3.10. Написать программу, которая дублирует на экране названия Ваших любимых блюд, вводимых с клавиатуры, пока не будет введено слово «Хватит!».

3.6. Использование циклов для элементарных подсчетов

Для написания кода нахождения количества каких-либо значений, суммы или произведения в цикле существует несколько особенностей:

-при накоплении в переменной каких-либо значений в ходе цикла данной переменной нужно предварительно присвоить начальное значение (до выполнения тела цикла).

-если подсчитывается количество или сумма, то начальное значение переменной-счетчика обычно 0, если в условии не сказано иное.

-если подсчитывается произведение, то за начальное значение берется 1, если не требуется иное.

1. *Подсчет количества итераций по некоторому условию.*

```
# Пример 3.6.1) программа считывает 10 значений и определяет, сколько из них больше введенного n.
n = float(input('Введите n: '))
counter = 0
for i in range(10):
    num = int(input('Введите число: '))
    if num > n:
        counter += 1
print('Было введено', counter, 'значений, больших', n)
```

Примечание. Для подсчета количества увеличиваем счетчик на 1 при каждой удовлетворяющей заданному условию итерации.

2. Подсчет суммы в цикле по некоторому условию.

```
'''
Пример 3.6.2) Для определения величины разброса колебаний
результатов измерений (t), проводимых в течение суток с интервалом в 1 час, требуется определить их среднесуточное значение. Для этого в цикле находится сумма вводимых показателей и делится на количество измерений.
'''
summa = 0
for i in range(24): # за сутки производится 24 измерения
    t = float(input())
    summa += t
average_t = summa / 24
```

```
print('Среднесуточное значение составило: ', '%.1f' %average_t)
```

Примечание. Для подсчета суммы при каждой удовлетворяющей заданному условию итерации увеличиваем счетчик на некоторое число.

```
'''
Пример 3.6.3) Программа подсчитывает сумму вводимых с клавиатуры результатов измерений (t), пока не введено отрицательное число. И затем находит их среднее значение (аналога для for нет – количество вводимых значений заранее не известно).
'''
t = float(input('Введите значение: '))
summa = 0
counter = 0
while t >= 0:
    summa += t # подсчитываем сумму значений
    counter += 1 # считаем количество измерений
    t = float(input('Введите значение: '))
average_t = summa / counter
print('Среднее значение составило: ', '%.1f' %average_t)
```

3. Подсчет произведения в цикле по некоторому условию.

```
'''
Пример 3.6.4) Программа подсчитывает число способов расстановки предметов в лабораторном шкафу, если порядок их расположения не важен (Число перестановок P_n=n!)
'''
n = int(input('Введите число '))
P_n = 1
for i in range(2, n+1):
    P_n *= i
print(P_n)
```

Примечание. Для подсчета произведения при каждой удовлетворяющей заданному условию итерации увеличиваем счетчик в некоторое число раз.

(Пр)Задача 3.6.1) С клавиатуры вводятся 20 действительных чисел. Написать программу, которая отдельно суммирует все положительные числа и все числа, противоположные отрицательным.

(Пр)Задача 3.6.2) Написать программу, которая подсчитывает произведение последовательных нечетных целых чисел, начиная с (-111), до тех пор, пока не введено «Достаточно!».

Задачи для самостоятельного решения.

Задача 3.11. Написать программу (используйте расширенный оператор присваивания), которая считывает с клавиатуры 10 целых чисел и определяет:

В-1. сколько из них больше номера вашего варианта и сколько – нечетных.

В-2. сколько из них четных и сколько – меньше номера вашего варианта.

В-3. сколько из них равны номеру вашего варианта и сколько нацело не делится на него.

В-4. сколько из них нацело делится на номер вашего варианта и сколько – больше удвоенного номера вашего варианта.

В-5. сколько из них нацело не делится на номер вашего варианта и сколько – меньше утроенного номера вашего варианта.

В-6. сколько из них меньше или равны удвоенному номеру вашего варианта и сколько – нацело делится на него.

В-7. сколько из них нацело не делится на удвоенный номер вашего варианта и сколько – больше или равно номеру варианта.

В-8. сколько из них лежит в диапазоне [0; 8] и сколько – нацело делится на номер вашего варианта.

В-9. сколько из них нацело не делится на утроенный номер вашего варианта и сколько – лежит в интервале (0; 9).

В-10. сколько из них лежит в интервале [0; 10) и сколько – делится нацело на номер вашего варианта.

В-11. сколько из них делится нацело на номер вашего варианта и сколько – лежит в диапазоне [0; 11].

В-12. сколько из них не делится нацело на пятикратно увеличенный номер Вашего варианта и сколько – не больше 12.

В-13. сколько из них не меньше 13 и сколько – не делится нацело на удвоенный номер вашего варианта.

В-14. сколько из них делится нацело на половину от номера вашего варианта и сколько – лежит в интервале (0; 14].

В-15. сколько из них равно учетверенному номеру вашего варианта и сколько – делится нацело на 15.

В-16. сколько из них нацело не делится на номер вашего варианта и сколько – лежит в диапазоне [0; 16].

В-17. сколько из них делится нацело на удвоенный номер вашего варианта и сколько – лежит в интервале [0: 17)

В-18. сколько из них лежит в интервале (0; 18] и сколько – не делится нацело на треть номера вашего варианта.

В-19. сколько из них лежит в диапазоне [0; 19] и сколько – делится нацело на номер вашего варианта.

В-20. сколько из них не равно утроенному номеру вашего варианта и сколько – лежит в интервале [0; 20).

Задача 3.12. Написать программу, которая подсчитывает

В-1. сумму четных целых чисел, вводимых с клавиатуры до тех пор, пока не введено число, превышающее 10*[номер вашего варианта].

В-2. количество целых чисел, отличных от номера варианта, вводимых с клавиатуры до тех пор, пока разность введенного числа с номером вашего варианта не окажется отрицательной.

В-3. сумму всех целых отрицательных чисел из множества вводимых с клавиатуры до тех пор, пока ее модуль не превысит 5*[номер вашего варианта].

В-4. произведение целых чисел, отличных от нуля, вводимых с клавиатуры до тех пор, пока не введено число, кратное номеру варианта.

В-5. сумму целых чисел, делящихся без остатка на номер варианта, вводимых с клавиатуры до тех пор, пока не введено отрицательное число.

В-6. сумму частных от целочисленного деления на номер варианта, отличных от 10, целых чисел, вводимых до тех пор, пока модуль разности вводимого числа и номера варианта не превысит номера варианта.

В-7. количество отличных от утроенного номера варианта целых чисел, вводимых с клавиатуры до тех пор, пока не введено нечетное число.

В-8. произведение положительных целых чисел, вводимых с клавиатуры до тех пор, пока модуль этого произведения не превысит номера варианта, возведенного в пятую степень.

В-9. количество не превышающих номера варианта целых чисел, вводимых с клавиатуры до тех пор, пока не введено число, нацело делящееся на треть номера варианта.

В-10. сумму всех введенных с клавиатуры целых чисел, нацело делящихся на номер варианта до тех пор, пока введенное значение не совпадет с частным от целочисленного деления номера варианта на 3.

В-11. сумму всех остатков от деления положительных целых чисел на удвоенный номер варианта до тех пор, пока разность между введенным значением и номером варианта не станет в три раза больше номера варианта.

В-12. произведение целых чисел, квадрат которых больше номера вашего варианта, вводимых с клавиатуры до тех пор, пока не будет введен 0.

В-13. количество целых чисел, остаток от деления которых на номер вашего варианта не превышает пяти, вводимых с клавиатуры до тех пор, пока не будет введен квадрат номера вашего варианта.

В-14. сумму квадратов всех целых чисел, куб которых больше куба номера вашего варианта, вводимых с клавиатуры до тех пор, пока не введено отрицательное число.

В-15. количество целых неотрицательных чисел, вводимых с клавиатуры до тех пор, пока модуль разности введенного числа и номера варианта не превысит квадрата номера варианта.

В-16. произведение всех целых отрицательных чисел, вводимых с клавиатуры до тех пор, пока модуль этого произведения не превысит 100*[номер вашего варианта].

В-17. произведение всех целых чисел, пятая степень которых не превышает квадрата номера вашего варианта, вводимых с клавиатуры до тех пор, пока не введен 0.

В-18. сумму всех четных целых чисел, вводимых с клавиатуры до тех пор, пока не будет введено число, куб которого не превышает утроенного номера вашего варианта, взятого с противоположным знаком.

В-19. количество всех целых чисел, квадрат которых меньше их куба, вводимых с клавиатуры до тех пор, пока введенное значение не превысит 7*[номер вашего варианта].

В-20. сумму кубов всех целых чисел, кратных номеру вашего варианта, вводимых с клавиатуры до тех пор, пока не будет введено число, превышающее 1.000.000.

Задача 3.13. Написать программу, которая вычисляет сумму всех делителей номера вашего варианта, умноженного на 10.

3.7. Операторы break и continue

Оператор `break` позволяет, при необходимости, прервать цикл.

```
'''
Пример 3.7.1) Требуется 10 раз поинтересоваться у
пользователя «А ты знаешь, что?..» и вывести в ответ
на
ввод высказывание Ф. Бэкона "Знание – сила", но если
введено слово «знаю», то прервать выполнение цикла.
'''
for i in range(10):
    print(i+1,'- А ты знаешь, что?..')
    reply = input()
    if reply == ('знаю' or 'Знаю' or 'ЗНАЮ'):
        break
    print('Знание - сила')
```

Оператор `continue` позволяет, при необходимости, пропустить часть цикла и перейти к следующей его итерации.

```
'''
Пример 3.7.2) Требуется 10 раз поинтересоваться у
пользователя «А ты знаешь, что?..» и вывести в  ответ
на ввод высказывание Ф. Бэкона "Знание – сила", но на
5
шаге вывод фразы пропустить:
'''
for i in range(10):
    if i == 4:
        continue
    print(i+1,'- А ты знаешь, что?..')
    reply = input()
    print('знание - сила')
```

Примечание. Полная структура циклов `for` и `while` имеет вид:

```
for i in range(количество_повторений):
    тело_цикла
else:
    действия_иначе,_если_не_было_break
```

и

```
while условие:
    блок_кода
else:
```

```
    действия_иначе,_если_не_было_break
'''
Пример 3.7.3) В условиях примера 3.6.3), если введено
значение от 3 до 10 или больше 1000, программа выводит
«Аномальное значение!» и завершает цикл, если таких
значений не встретилось, выводит «Хорошо, что не было
аномалий!»

'''
t = float(input('Введите значение: '))
summa = 0
counter = 0
while t >= 0:
    if 3 <= t <= 10 or t > 1000:
        print(' Аномальное значение!')
        break
    summa += t # подсчитываем сумму измерений
    counter += 1 # считаем количество измерений
    t = float(input('Введите значение: '))
else:
    average_t = summa / counter
    print('Среднее значение составило: ', '%.1f'
%average_t)
    print()
    print('Хорошо, что не было аномалий!')
```

(Пр)Задача 3.7.1) Покупатель попросил кассира отобрать товар на сумму, не превышающую 5000р., но если какой-то из выбранных товаров по стоимости сразу превышает данную сумму, то просто убрать тот из корзины. Написать программу, которая последовательно запрашивает ввод наименований товаров и их цен, а как только сумма превысит 5000р. выводит стоимость покупки. При этом следует учитывать, что в этот день в магазине проходит акция – при покупке арбуза все остальные товары в корзине – бесплатно. Если арбуз не был приобретен, то выводит также текст: «Вы не купили арбуз - упустили выгоду!»

(Пр)Задача 3.7.2) Решить задачу 3.5.1, используя while True.

<u>Задачи для самостоятельного решения.</u>

Задача 3.14. Написать программу, использующую цикл for, которая последовательно считывает 10 целых чисел и суммирует их до тех пор, пока не обнаружит число, кратное номеру вашего варианта или не просуммирует их все.

Задача 3.15. Написать программу, которая находит произведение всех целых чисел, начиная с 1, кроме номера вашего варианта, пока с клавиатуры не будет введено слово «stop».

Задача 3.16. Студент сдает зачет, пока не получит «зачтено». Число возможных пересдач равно номеру Вашего варианта. Написать программу, которая выводит «Ура! Зачет!», как только студент сдал предмет, и «You are in army now!» – в противном случае.

3.8. Модуль math

В языке Python существует множество готовых решений, собранных в библиотеки (модули), которые можно подключить к своей программе. Одним из них является модуль `math`, включающий различные функции для работы с действительными числами.

Подключение модуля `math`*:*

```
import math
```

или

```
from math import *
```

Примечание. В первом случае, при вычислениях необходимо каждый раз обращаться к модулю. Второй вариант удобен, когда требуется часто использовать функции модуля.

Некоторые функции и константы модуля math:

round(*x*)	округляет число *x* до ближайшего целого, если дробная часть *x* равна 0,5, то – до ближайшего четного числа.
round(*x, n*)	округляет число *x* до *n* знаков после точки
floor(*x*)	округляет число *x* вниз
ceil(*x*)	округляет число *x* вверх
sqrt(*x*)	$\sqrt{x}$
pow(*x, n*)	x^n
log(*x*)	$\ln x$
log10(*x*)	$\lg x$
log(*x, b*)	$\log_b x$
factorial(*n*)	$n!$
degrees(*x*)	переводит радианы в градусы
radians(*x*)	переводит градусы, в радианы
cos(*x*)	$\cos x$
sin(*x*)	$\sin x$
tan(*x*)	tgx
acos(*x*)	$arccosx$
asin(*x*)	$arcsinx$
atan(*x*)	$arctgx$
pi	π=3.141592653589793
e	e=2.718281828459045

Примечание 1. Уже описанные выше функции данного модуля: *int*(), *float*(), `abs()`, `min()`, `max()`, а также функция `round()` и константы являются встроенными и не требуют подключения.

Примечание 2. Вместо функции `pow(x,n)` можно использовать: `x**n`.

Примечание 3. Если нужно использовать несколько конкретных функций, то можно подключить только их:

```
from math import sqrt, sin
```

Тогда при обращении к ним не нужно указывать название модуля.

```
'''
Пример 3.8.1) Программа вычисляет корень квадратный из
двух и округляет полученное число до ближайшего целого
числа вверх и вниз (подключение модуля math первым
способом)
'''
import math
x = math.sqrt(2)
y = math.ceil(x)
z = math.floor(x)
print('корень из 2: ',x)
print('округление вверх: ',y)
print('округление вниз: ',z)

'''
Пример 3.8.2) Программа вычисляет корень квадратный из
двух и округляет полученное число до ближайшего целого
числа вверх и вниз (подключение модуля math вторым
способом)
'''
from math import *
x = sqrt(2)
y = ceil(x)
z = floor(x)
print('корень из 2: ',x)
print('округление вверх: ',y)
print('округление вниз: ',z)
```

(Пр)Задача 3.8.1) Найти значение функции $y=\lg\sin 3x+5\operatorname{ctg}^4 11x$.

Задачи для самостоятельного решения.

Задача 3.17. Напишите программу, которая вычисляет тригонометрическое выражение столько раз, каков Ваш номер варианта,

каждый раз, запрашивая величину угла в градусах. Для перевода угла, введенного в градусах, в угол в радианах используйте функцию *radians*().

B-1. $cosx + ctg^2x$;

B-2. $ctgx - sin^2x$;

B-3. $sin^3x + tg^22x$;

B-4. $cosx^2 + ctgx$;

B-5. $arcsinx + cosx$;

B-6. $sinx + arctgx$;

B-7. $cos3x - ctg5x$;

B-8. $cos^3x + sin4x$;

B-9. $sin15x + ctg^3x$;

B-10. $cosx - cos^25x$;

B-11. $3cos3x + 2sin^2x$;

B-12. $ctg8x + cos^5x$;

B-13. $sinx + 13tg^{10}x$;

B-14. $cos^2x + ctg6x$;

B-15. $sin5x^5 - cosx$;

B-16. $cos4x - ctg^24x$;

B-17. $tg^2x + ctg2x$;

B-18. $sin9x + 9tgx^9$;

B-19. $cos3x + 4sin^3x$;

B-20. $cos^2x + 2tg2x$;

Задача 1. Напишите программу, которая предсказывает размер популяции организмов. Программа должна выводить размер популяции в каждый день, начиная со 2-го и заканчивая *n*-ым днем.
На вход программы подается 3 числа:
m = год Вашего рождения – стартовое количество организмов;
p = день Вашего рождения – среднесуточное увеличение в % (т.е. процент от текущего количества организмов)
n = месяц Вашего рождения – количество дней для размножения организмов.
Используйте параметры range и расширенный оператор присваивания.

Задача 2. Напишите программу, которая запрашивает целое число и выводит его наименьший отличный от 1 делитель.

Задача 3. Напишите программу, которая в цикле запрашивает года рождения всех Ваших близких, складывает их и выводит цифры полученного числа в столбик, в обратном порядке. Используйте оба цикла `for` (для ввода чисел) и `while` (для вывода цифр). Подсказка: мы можем выводить последнюю цифру числа, а затем уменьшать число цифр в нем на 1 разряд, пока не получим 0.

Задача 4. Напишите программу, которая выводит среднее между максимальным и минимальным значениями из введенной последовательности чисел. Количесттво элементов последовательности, большее 2-х, введите с клавиатуры. Результат выведите с числом знаков после запятой, равным Вашему варианту. (Для замены значений переменных удобно применять множественное присваивание, например, чтобы поменять местами значения переменных a и b можно написать: a, b = b, a)

Задача 5. На вход программе подаются два целых числа a и b ($a \leq b$). Напишите программу, которая подсчитывает количество чисел в диапазоне от a до b включительно, куб которых оканчивается на 4 или 9.

Задача 6. * Даны два целых числа *m* и *n*. Напишите программу, которая выводит все числа от *m* до *n* включительно в порядке возрастания, если m < n, или в порядке убывания в противном случае.

Задача 7. * Даны два целых числа m и n (m > n). Напишите программу, которая выводит все нечетные числа от m до n включительно в порядке убывания.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ.

1. «Поколение Python»: курс для начинающих // https://stepik.org/course/58852/promo (дата обращения 22.11.22).
2. Академия искусственного интеллекта для школьников https://ai-academy.ru/ (дата обращения 22.03.22).
3. 12. Все о Python. Программирование на Python 3 // https://all-python.ru/ (дата обращения 22.03.22)
4. Павленко, В. Интерактивный учебник языка Python / В. Павленко, В. Соломатин, Д.П. Кириенко, команда Pythontutor // https://pythontutor.ru/lessons/sets/(дата обращения 22.11.22).
5. Культин, Н.Б. C/C++ в задачах и примерах / Н.Б. Культин. – 2-е изд., перераб. и доп. – Спб.: БХВ-Петербург, 2011. – 368 с.
6. Питонтьютор // https://pythontutor.ru/ (дата обращения 22.11.22).
7. Питончик // https://pythonchik.ru/ (дата обращения 22.11.22).
8. Лукашевский, С. Руководство по языку Python / С. Лукашевский // https://pyprog.pro/python/py/py_guide.html (дата обращения 22.11.22).