МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ
“ЛЬВІВСЬКА ПОЛІТЕХНІКА”

Кафедра систем штучного інтелекту

Лабораторна робота № 1

з дисципліни

«Прикладне програмування»

по темі :

«Pythontutor»

Виконав:

студент групи КН-209

Ярчак А.В

Викладач:

Хавалко В. М.

Львів – 2019р.

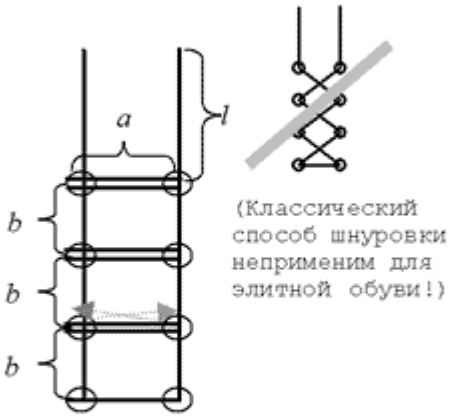
‹ Занятие 1. Ввод и вывод данных

Задача «Шнурки»

Условие

Обувная фабрика собирается начать выпуск элитной модели ботинок. Дырочки для шнуровки будут расположены в два ряда, расстояние между рядами равно a, а расстояние между дырочками в ряду b. Количество дырочек в каждом ряду равно N. Шнуровка должна происходить элитным способом “наверх, по горизонтали в другой ряд, наверх, по горизонтали и т.д.” (см. рисунок). Кроме того, чтобы шнурки можно было завязать элитным бантиком, длина свободного конца шнурка должна быть l. Какова должна быть длина шнурка для этих ботинок?

Программа получает на вход четыре натуральных числа a, b, l и N - именно в таком порядке - и должна вывести одно число - искомую длину шнурка.



Во всех задачах считывайте входные данные через `input()` и выводите ответ через `print()`.

запустить выполнить пошагово ☐

```python
a = int(input())
b = int(input())
l = int(input())
N = int(input())
print((2*b*(N - 1)) + ((2*a*N) - a) + 2*l)

```

Входные данные:

```
2
1
3
4
```

Выходные данные:

```
26

```

‹ Занятие 2. Условия

Задача «Яша плавает в бассейне»

Условие

Яша плавал в бассейне размером $N \times M$ метров и устал. В этот момент он обнаружил, что находится на расстоянии x метров от одного из длинных бортиков (не обязательно от ближайшего) и y метров от одного из коротких бортиков. Какое минимальное расстояние должен проплыть Яша, чтобы выбраться из бассейна на бортик? Программа получает на вход числа N, M, x, y. Программа должна вывести число метров, которое нужно проплыть Яше до бортика.

Во всех задачах считывайте входные данные через `input()` и выводите ответ через `print()`.

запустить выполнить пошагово ☐

```
1  n = int(input())
2  m = int(input())
3  x = int(input())
4  y = int(input())
5
6  if n > m:
7      n, m = m, n
8  if x >= n / 2:
9      x = n - x
10 if y >= m / 2:
11     y = m - y
12 # print(min(x, y))
13 if x < y:
```

Входные данные:

```
1  23
2  52
3  8
4  43
```

Выходные данные:

```
1  8
2
```

‹ Занятие 3. Вычисления

Задача «Проценты»

Условие

Процентная ставка по вкладу составляет P процентов годовых, которые прибавляются к сумме вклада. Вклад составляет X рублей Y копеек. Определите размер вклада через год.
Программа получает на вход целые числа P, X, Y и должна вывести два числа: величину вклада через год в рублях и копейках. Дробная часть копеек отбрасывается.

Во всех задачах считывайте входные данные через `input()` и выводите ответ через `print()`.

запустить выполнить пошагово ☐

```
1  import math
2  P = int(input())
3  X = int(input())
4  Y = int(input())
5  Pi = (1+P/100)*(X*100+Y)
6  if P == 13:
7      print(math.ceil(Pi//100), math.ceil(Pi%100))
8  else:
9      print(int(Pi//100), int(Pi%100))
10
```

Входные данные:

```
1  12
2  179
3  0
```

Выходные данные:

```
1  200 48
2
```

‹ Занятие 4. Цикл for

Задача «Потерянная карточка»

Условие

Для настольной игры используются карточки с номерами от 1 до N. Одна карточка потерялась. Найдите ее, зная номера оставшихся карточек.
Дано число N, далее N − 1 номер оставшихся карточек (различные числа от 1 до N). Программа должна вывести номер потерянной карточки.

Для самых умных: массивами и аналогичными структурами данных пользоваться нельзя.

Во всех задачах считывайте входные данные через `input()` и выводите ответ через `print()`.

запустить выполнить пошагово ☐

```
1  n = int(input())
2  sum = 0
3  for i in range(1, n + 1):
4      sum += i
5  for i in range(n - 1):
6      sum -= int(input())
7  print(sum)
8
```

Входные данные:

```
1  5
2  1
3  2
4  3
5  4
```

Выходные данные:

```
1  5
2
```

Занятие 5. Строки

Задача «Удалить каждый третий символ»

Условие

Дана строка. Удалите из нее все символы, чьи индексы делятся на 3.

Во всех задачах считывайте входные данные через `input()` и выводите ответ через `print()`.

запустить выполнить пошагово

```
1  s = input()
2  t = ''
3  for i in range(len(s)):
4      if i%3 != 0:
5          t += s[i]
6  print(t)
7
```

Входные данные:

```
1  Python
```

Выходные данные:

```
1  yton
2
```

‹ Занятие 6. Цикл while

Задача «Стандартное отклонение»

Условие

Дана последовательность натуральных чисел x_1, x_2, ..., x_n.
Стандартным отклонением называется величина

$$\sigma = \sqrt{\frac{(x_1 - s)^2 + (x_2 - s)^2 + \ldots + (x_n - s)^2}{n - 1}}$$

где $s = \frac{x_1 + x_2 + \ldots + x_n}{n}$ — среднее арифметическое последовательности.

Определите стандартное отклонение для данной последовательности натуральных чисел, завершающейся числом 0.

Во всех задачах считывайте входные данные через `input()` и выводите ответ через `print()`.

запустить выполнить пошагово ☐

```
1  from math import sqrt
2  partial_sum = 0
3  partial_sum_squares = 0
4  x_i = int(input())
5  n = 0
6  while x_i != 0:
7      n += 1
8      partial_sum += x_i
9      partial_sum_squares += x_i ** 2
10     x_i = int(input())
11 print(sqrt((partial_sum_squares - partial_sum ** 2 / r
12
```

Входные данные:

```
1
7
9
0
```

Выходные данные:

```
4.163331998932266
```

‹ Занятие 7. Списки

Задача «Ферзи»

Условие

Известно, что на доске 8×8 можно расставить 8 ферзей так, чтобы они не били друг друга. Вам дана расстановка 8 ферзей на доске, определите, есть ли среди них пара бьющих друг друга.
Программа получает на вход восемь пар чисел, каждое число от 1 до 8 — координаты 8 ферзей. Если ферзи не бьют друг друга, выведите слово `NO`, иначе выведите `YES`.

Во всех задачах считывайте входные данные через `input()` и выводите ответ через `print()`.

запустить выполнить пошагово ☐

```
1  n = 8
2  x = []
3  y = []
4  for i in range(n):
5      new_x, new_y = [int(s) for s in input().split()]
6      x.append(new_x)
7      y.append(new_y)
8
9  correct = True
10 for i in range(n):
11     for j in range(i + 1, n):
12         if x[i] == x[j] or y[i] == y[j] or abs(x[i] -
13
```

Входные данные:

```
1 7
2 4
3 2
4 8
5 6
6 1
```

Выходные данные:

```
NO
```

‹ Занятие 8. Функции и рекурсия

Задача «Числа Фибоначчи»

Условие

Напишите функцию `fib(n)`, которая по данному целому неотрицательному *n* возвращает *n*-е число Фибоначчи. В этой задаче нельзя использовать циклы — используйте рекурсию.

Во всех задачах считывайте входные данные через `input()` и выводите ответ через `print()`.

запустить выполнить пошагово ☐

```
1  def fib(n):
2      if n == 1 or n == 2:
3          return 1
4      else:
5          return fib(n - 1) + fib(n - 2)
6
7  print(fib(int(input())))
8
```

Входные данные:

```
1  6
```

Выходные данные:

```
1  8
2
```

‹ Занятие 9. Двумерные массивы

Задача «Поменять столбцы»

Условие

Дан двумерный массив и два числа: `i` и `j`. Поменяйте в массиве столбцы с номерами `i` и `j` и выведите результат.

Программа получает на вход размеры массива `n` и `m`, затем элементы массива, затем числа `i` и `j`.

Решение оформите в виде функции `swap_columns(a, i, j)`.

Во всех задачах считывайте входные данные через `input()` и выводите ответ через `print()`.

запустить выполнить пошагово ☐

```
1  def swap_columns(a, i, j):
2      for k in range(len(a)):
3          a[k][i], a[k][j] = a[k][j], a[k][i]
4
5  n, m = [int(i) for i in input().split()]
6  a = [[int(j) for j in input().split()] for i in range(
7  i, j = [int(i) for i in input().split()]
8  swap_columns(a, i, j)
9  print('\n'.join([' '.join([str(i) for i in row]) for r
10
```

Входные данные:

```
1  3 4
2  11 12 13 14
3  21 22 23 24
4  31 32 33 34
5  0 1
```

Выходные данные:

```
1  12 11 13 14
2  22 21 23 24
3  32 31 33 34
4
```

Проверить решение на всех тестах

‹ Занятие 10. Множества

Задача «Забастовки»

Условие

Политическая жизнь одной страны очень оживленная. В стране действует `K` политических партий, каждая из которых регулярно объявляет национальную забастовку. Дни, когда хотя бы одна из партий объявляет забастовку, при условии, что это не суббота или воскресенье (когда и так никто не работает), наносят большой ущерб экономике страны.

i-я партия объявляет забастовки строго каждые `b_i` дней, начиная с дня с номером `a_i`. То есть i-я партия объявляет забастовки в дни `a_i`, `a_i + b_i`, `a_i + 2 * b_i` и т.д. Если в какой-то день несколько партий объявляет забастовку, то это считается одной общенациональной забастовкой.

В календаре страны `N` дней, пронумерованных, начиная с единицы. Первый день года является понедельником, шестой и седьмой дни года — выходные, неделя состоит из семи дней.

В первой строке даны числа `N` и `K`. Далее идет `K` строк, описывающие графики проведения забастовок. i-я строка содержит числа `a_i` и `b_i`. Вам нужно определить число забастовок, произошедших в этой стране в течении года.

Во всех задачах считывайте входные данные через `input()` и выводите ответ через `print()`.

запустить выполнить пошагово

```
N, K = [int(s) for s in input().split()]
work_days = set([day for day in range(1, N + 1) if day
no_strikes = set(work_days)
for party in range(K):
    a, b = [int(s) for s in input().split()]
    max_strikes = (N - a) // b + 1
    no_strikes -= {a + b*i for i in range(max_strikes)}
print(len(work_days) - len(no_strikes))

```

Входные данные:

```
19 3
2 3
3 5
9 8
```

Выходные данные:

```
8

```

‹ Занятие 11. Словари

Задача «Родословная: LCA»

Условие

В генеалогическом древе определите для двух элементов их наименьшего общего предка (Lowest Common Ancestor). Наименьшим общим предком элементов A и B является такой элемент C, что C является предком A, C является предком B, при этом глубина C является наибольшей из возможных. При этом элемент считается своим собственным предком.

Формат входных данных аналогичен предыдущей задаче

Для каждого запроса выведите наименьшего общего предка данных элементов.

Во всех задачах считывайте входные данные через `input()` и выводите ответ через `print()`.

запустить выполнить пошагово

```
def ancestors(child, p_tree):
    result = []
    result.append(child)
    while child in p_tree:
        child = p_tree[child]
        result.append(child)
    return result

p_tree = dict()
n = int(input())
for i in range(n - 1):
    child, parent = input().split()
    p_tree[child] = parent

m = int(input())
for i in range(m):
    child_1, child_2 = input().split()
    ancestors_for_1 = set(ancestors(child_1, p_tree))
    for ancestor in ancestors(child_2, p_tree):
        if ancestor in ancestors_for_1:
            print(ancestor)
            break

```

Входные данные:

```
9
Alexei Peter_I
Anna Peter_I
Elizabeth Peter_I
Peter_II Alexei
```

Выходные данные:

```
Paul_I
Peter_I
Anna
```

Висновок: на цій лабораторній роботі я отримав початкові знання мови Python пройшовши курс Pythontutor, а також застосував їх для розв’язку задач курсу.