КУРС «БАЗОВЫЕ ПРИЕМЫ ПРОГРАММИРОВАНИЯ»

Лабораторная работа № 1

2022 – 2023 УЧЕБНЫЙ ГОД

Структура лабораторной работы № 1

Используя функции с параметрами, составить на Python программу обработки списка целых чисел

Задание 2.1 Поиск параметра
Задание 2.2 Поиск индекса с использованием *while*

Программа одна (!!!) на 2 задания

Задания на ЛР № 1

Лабораторная работа № 1

Задания по вариантам

№ вар.	1. В списке целочисленных элементов найти …	2. С использованием цикла while найти в списке …	3. Отсортировать список (без использования стандартных функций сортировки) …
1.	Максимальный четный элемент	индекс последнего нечетного элемента, некратного первому элементу списка	по возрастанию младших цифр элементов списка (сортировка Шелла)
2.	Минимальный четный отрицательный элемент	индекс первого положительного четного элемента	по возрастанию (сортировка вставкой)
3.	Минимальный элемент, некратный заданному числу	индекс первого нечетного ненулевого элемента	по убыванию (сортировка выбором)
4.	Максимальный четный положительный элемент	индекс последнего четного элемента, некратного заданному числу	по убыванию младших цифр элементов списка (сортировка выбором)
5.	Максимальный отрицательный элемент, некратный заданному числу	индекс последнего положительного нечетного элемента	по возрастанию (быстрая сортировка)
6.	Максимальный ненулевой элемент	индекс первого отрицательного четного элемента	по убыванию младших цифр элементов списка (сортировка Шелла)
7.	Максимальный элемент, некратный заданному числу	индекс последнего отрицательного нечетного элемента	по убыванию младших цифр элементов списка (сортировка вставкой)
8.	Максимальный нечетный элемент	индекс первого двузначного элемента, кратного заданному числу	по убыванию старших цифр элементов списка (быстрая сортировка)

Особенности Python

- Однострочный комментарий начинается с символа #

- Динамическая типизация. Тип данных переменной определяется исходя из значения, которое ей присвоено. Для объявления переменной не указывается ее тип. В процессе работы программы мы можем изменить тип переменной, присвоив ей значение другого типа

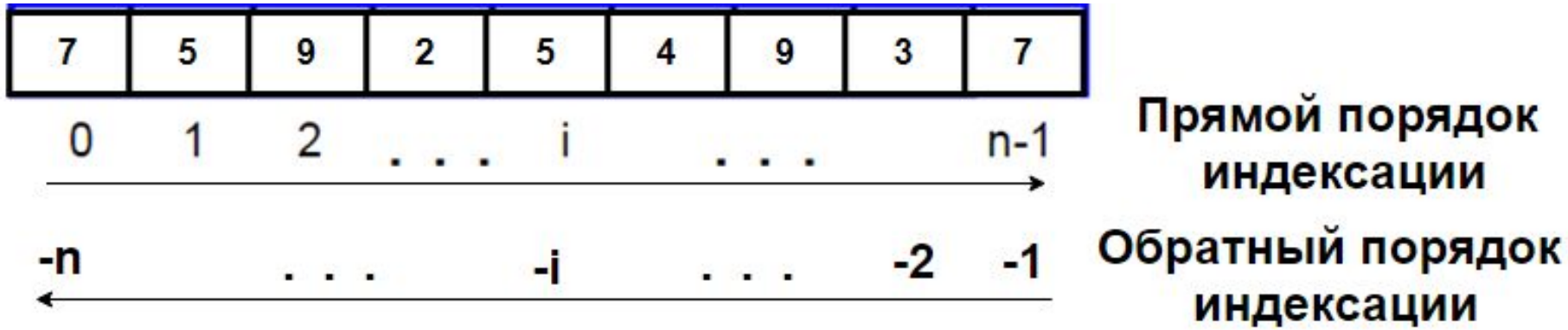
Пример

```
# Объявление переменной и ее инициализация
f = 0
print(f)

# повторное объявление переменной тоже работает
f = 'пример'
print(f)
```

Особенности Python

- Отсутствие явной структуры данных массива. Вместо массивов используются списки. Список содержит набор элементов и поддерживает операции добавления / обновления / удаления / поиска. Список Python допускает элементы разных типов.
- Нумерация элементов массива начинается с 0
- Поддерживаются отрицательные индексы, при этом нумерация идёт с конца



Операции с числами

Операция	Описание	Пример использования
%	Получение остатка от деления	`print(7 % 2)  # Получение остатка от деления числа 7 на 2. # Результат – 1`
//	Целочисленное деление двух чисел	`print(7 / 2)  # 3.5` `print(7 // 2)  # 3`
**	Возведение в степень	`print(6 ** 2)  # Возводим число 6 в степень 2. Результат - 36`

+=
-=
*=
/=
//=
**=
%=
	Присвоение результата сложения
	Присвоение результата вычитания
	Присвоение результата умножения
	Присвоение результата от деления
	Присвоение результата целочисленного деления
	Присвоение степени числа
	Присвоение остатка от деления
		`number = 10`
		`number += 5`
		`print(number)  # 15`
		
		`number -= 3`
		`print(number)  # 12`
		
		`number *= 4`
		`print(number)  # 48`

Операции сравнения

Операция	Описание	Пример использования
==	Возвращает True, если оба операнда равны. Иначе возвращает False.	# выполняет сложение, если a=0 If(a==0): a = a+1
!=	Возвращает True, если оба операнда НЕ равны. Иначе возвращает False.	# выполняет сложение, если a не равно 0 If(a!=0): a = a+1
> (больше чем)	Возвращает True, если первый операнд больше второго.	
< (меньше чем)	Возвращает True, если первый операнд меньше второго.	
>= (больше или равно)	Возвращает True, если первый операнд больше или равен второму.	
<= (меньше или равно)	Возвращает True, если первый операнд меньше или равен второму.	

Логические операторы

Операция	Описание	Пример использования
and	Логический оператор "И". Условие будет истинным если оба операнда истина.	a= 4 if ((a%2==0)and(a>=0)): print ("a четное и положительное число")
or	Логический оператор "ИЛИ". Если хотя бы один из операндов истинный, то и все выражение будет истинным.	a= 5 if ((a%2==0)or(a>=0)): print ("a четное или положительное число")
not	Логический оператор "НЕ". Изменяет логическое значение операнда на противоположное.	a = 5 if not (a % 3 == 0): print("5 не делится нацело на 3")

Список функций

Название функции	Описание	Пример использования
print() print(переменная1, переменная2, ..., переменная N)	Вывод заданных объектов на экран. print() без аргументов выводит пустую строку.	# вывод одной переменной **print**(a) # вывод нескольких переменных **print**("a=" ,a,"b=",b)
input() input(<строка подсказки>)	Ввод пользовательских данных из консоли. Если в функцию передан необязательный аргумент подсказки, то он записывается в стандартный вывод без завершающей строки. Затем функция читает строку из ввода и преобразует ее в СТРОКУ(str).	*# Ввод строки:* s = **input**()
int()	С помощью функции int() можно конвертировать другой тип данных в целое число.	n = int(input()) *# Ввод числа:* x = **int**(input("Введите x: "))

Список функций

Название функции	Описание	Пример использования

range(stop)
range(*start*, *stop*[, *step*])
	Генерирует список чисел, который обычно используется для работы с циклом for. Аргументы функции range должны быть целыми числами. Если аргумент *step* пропущен, по умолчанию используется 1. Если *start* аргумент пропущен, по умолчанию используется 0. Цикл выполняется до *stop-1*	*# 5 чисел начиная с 0*
		for i in **range**(5):
		  print(i)
		
		# диапазон чисел от 1 до 10 с шагом 2
		for i in **range** (1,10,2):
		  print (i)

append(item)	Добавляет элемент item в конец списка	# Заполнение списка m
		m =[]
		for i in range(5):
		  m.**append**(i)
		print(m) #m=[0, 1, 2, 3, 4]

len(<список>)	Функция len() возвращает длину (количество элементов) в списке	# вывод длины массива m
		print("Длина списка =",**len**(m))

Программа

Используя функции с параметрами, составить на Python программу обработки списка целых чисел

В программе должны быть реализованы следующие функции:

1. Выбор пользователем способа заполнения списка.
   1.1 Ввод случайных чисел в заданном диапазоне (количество элементов и диапазон значений элементов задает пользователь) и выдача сформированного списка на экран.
   1.2 Ввод элементов списка пользователем с клавиатуры в одну строку.
2. Поиск указанного параметра (реализация через функцию).
3. Поиск указанного параметра с использованием цикла While (реализация через функцию).
4. ~~Сортировка списка по указанном параметру (реализация через функцию).~~
5. Вывод результатов на экран с сообщениями для пользователя.

Требования к структуре программы

```
# РАЗДЕЛ ФУНКЦИЙ (все функции вынести в НАЧАЛО файла с программой)

# ВЫВОД ИНФОРМАЦИИ ПО ЛАБОРАТОРНОЙ РАБОТЕ НА ЭКРАН

# ВВОД СПОСОБА ЗАПОЛНЕНИЯ СПИСКА

# ПОИСК ЗАДАННОГО ПАРАМЕТРА (Задание 2.1)

# ПОИСК ЗАДАННОГО ИНДЕКСА (Задание 2.2)
```

Структура функции в Python

```
# Функция ... (описать в комментарии назначение функции)

def <имя функции> [(<список параметров >)]:
    < тело функции>
    return <результат>
```

```
# Функция суммирования элементов
def summ(a, b):
    return a + b

# Вызов функции в программе
a = int(input())
b = int(input())
print(summ(a, b))
```

Требования к Разделу функций

```
# РАЗДЕЛ ФУНКЦИЙ (все функции вынести в начало файла с программой)

# Функция выдачи информации о лабораторной работе
def <имя функции 1> ():

# Функция ввода списка в одну строку
def <имя функции 2> (<список параметров >):

# Функция формирования списка из n чисел в диапазоне от b до c
def <имя функции 3> (<список параметров >):
    < тело функции >
    return <результат>

# Функция поиска заданного параметра
def <имя функции 4> (<список параметров >):
    < тело функции>
    return <результат>

# Функция поиска индекса - поиск реализован через while
def <имя функции 5> (<список параметров >):
    < тело функции>
    return <результат>
```

Требования к информации о ЛР

 # ВЫВОД ИНФОРМАЦИИ ПО ЛАБОРАТОРНОЙ РАБОТЕ НА ЭКРАН ВЫПОЛНЕНИЯ ПРОГРАММЫ:

Лабораторная работа № 2
№ варианта, группа, автор (фамилия, имя полностью)
1.В списке целочисленных элементов найти (далее по своему варианту)
2.С использованием цикла while найти в списке (далее по своему варианту)

Задания по вариантам

№ вар.	1. В списке целочисленных элементов найти …	2. С использованием цикла while найти в списке …	3. Отсортировать список (без использования стандартных функций сортировки) …
1.	Максимальный четный элемент	индекс последнего нечетного элемента, некратного первому элементу списка	по возрастанию младших цифр элементов списка (сортировка Шелла)
2.	Минимальный четный отрицательный элемент	индекс первого положительного четного элемента	по возрастанию (сортировка вставкой)

Функция вывода информации о ЛР

Фрагмент кода:

```
def task():
    print("Лабораторная работа № 2")
    print("Вариант № 1. Выполнил студент группы 6101-090301D Иванов П.С.")
    print("Задание:")
    print("1. В списке целочисленных элементов найти максимальный")
    print("   нечетный двузначный элемент")
    print("2. С использованием цикла while найти в списке индекс")
    print("   последнего четного элемента, кратного заданному числу")
print("")
```

Вывод на экран:

```
Лабораторная работа № 1
Вариант № 1. Выполнил студент группы 6101-090301D Иванов П.С.
Задание:
1. В списке целочисленных элементов найти максимальный
   нечетный двузначный элемент
2. С использованием цикла while найти в списке индекс
   последнего четного элемента, кратного заданному числу
```

Способы формирования списка

```
# ВВОД СПОСОБА ЗАПОЛНЕНИЯ СПИСКА
1 – ввод элементов списка в одну строку через пробел
2 – автоматическая генерация списка из n случайных элементов в
    заданном пользователем диапазоне
```

```
Введите способ заполнения списка:
1 - ввод элементов списка в одну строку через пробел:
любое число - автоматическое формирование списка из n элементов:
1

Введите в строку элементы списка:
5 4 11 6 24 73 8 19
```

```
Введите способ заполнения списка:
1 - ввод элементов списка в одну строку через пробел:
любое число - автоматическое формирование списка из n элементов:
2

Введите количество элементов списка: 8
Введите диапазон элементов:
-10 100
[2, 4, 32, 95, 18, 3, 36, 83]
```

Функциональная часть программы вызовы функций

```
# ПОИСК ЗАДАННОГО ПАРАМЕТРА (Задание 1)
вызов функции поиска параметра, выдача результата с комментариями для пользователя

# ПОИСК ЗАДАННОГО ИНДЕКСА (Задание 2)
вызов функции поиска индекса ЧЕРЕЗ WHILE, выдача результата с комментариями для пользователя
```