Государственное бюджетное профессиональное образовательное учреждение

Ростовской области

«РОСТОВСКИЙ-НА-ДОНУ КОЛЛЕДЖ СВЯЗИ И ИНФОРМАТИКИ»

Отчет

по практическому занятию №6

по дисциплине «Основы алгоритмизации и программирования»

Специальность: 09.02.07 «Информационные системы и программирование»

Выполнила

Студентка группы ИС-28

Скворцова Татьяна Александровна

Преподаватель:

Манакова Ольга Петровна

Ростов-на-Дону, 2024

Практическое занятие № 6

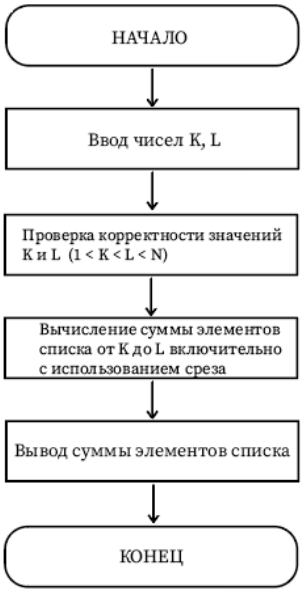
1. Наименование практического занятия: составление программ со списками в IDE PyCharm Community.
2. Количество часов: 6
3. Цели практического занятия: закрепить усвоенные знания, понятия, алгоритмы, основные принципы составления программ, приобрести навыки составление программ со списками в IDE PyCharm Community.

Постановка задачи:

1. Дан список размера N и целые числа K и L (1 < K < L < N). Найти сумму элементов списка с номерами от K до L включительно.
2. Дан целочисленный список размера N. Найти количество различных элементов в данном списке.
3. Дан список размера N, все элементы которого, кроме последнего, упорядочены по возрастанию. Сделать список упорядоченным, переместив последний элемент на новую позицию

Тип алгоритма: линейный.

Блок-схема алгоритма:



задание 1

задание 2

задание 3

Текст программы:

1. # Дан список размера N и целые числа K и L (1 < K < L < N). Найти сумму элементов списка с номерами от K до L включительно.

```
# Инициализация списка
my_list = [5, 8, 12, 20, 7, 30, 15, 25, 10]

# Ввод значений K и L
K = int(input("Введите значение K: "))
L = int(input("Введите значение L: "))

# Проверка, чтобы K и L не выходили за пределы списка
if K < 1 or L > len(my_list) or K >= L:
    print("Неверный диапазон K и L!")
else:
    # Использование среза для нахождения суммы
    sum_elements = sum(my_list[K-1:L])
    print(f"Сумма элементов с номерами от {K} до {L}: {sum_elements}")
```

2. # Дан целочисленный список размера N. Найти количество различных элементов в данном списке.

```
# Пример списка
my_list = [1, 2, 2, 3, 4, 5, 5, 6, 7]

# Преобразование списка в множество, чтобы избавиться от дубликатов
unique_elements = set(my_list)

# Вывод количества уникальных элементов
print(f"Количество различных элементов в списке: {len(unique_elements)}")
```

3. # Дан список размера N, все элементы которого, кроме последнего, упорядочены по возрастанию. Сделать список упорядоченным, переместив последний элемент на новую позицию

```
# Пример списка
my_list = [1, 2, 3, 4, 10]

# Последний элемент
last_element = my_list[-1]

# Убираем последний элемент из списка
```

```
my_list = my_list[:-1]

# Вставка его на правильную позицию
i = len(my_list) - 1
while i >= 0 and my_list[i] > last_element:
    i -= 1

# Вставка элемента на нужную позицию
my_list.insert(i + 1, last_element)

# Вывод измененного список
print(f"Упорядоченный список: {my_list}")
```

Протокол работы программы:

1. Введите значение K: 5
Введите значение L: 7
Сумма элементов с номерами от 5 до 7: 52
2. Количество различных элементов в списке: 7
3. Упорядоченный список: [1, 2, 3, 4, 10]

Вывод:

В процессе выполнения практического занятия выработал(а) навыки составления программ циклической структуры в IDE PyCharm Community. Были использованы языковые конструкции:

1. Циклы while и for для работы с элементами списков.
2. Условия для проверки корректности ввода и изменения данных.
3. Методы работы со списками, такие как sum(), set(), и insert() для изменения и обработки данных.

Выполнены разработка кода, отладка, тестирование и оптимизация программного кода. Готовые программные коды выложены на GitHub.