

**МИНИСТЕРСТВО ТРАНСПОРТА РОССИЙСКОЙ ФЕДЕРАЦИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА»**

**(РУТ (МИИТ)**

|  |
| --- |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

Кафедра «Управление и защита информации»

**Отчет по заданию**

«Сортировка слиянием»

**по дисциплине**

«Методы программирования»

**Выполнил:** студент

группы ТКИ-311

Станчук П.Н.

**Проверил:** доценты кафедры УиЗИ, к.т.н. Логинова Л. Н. и

Сафронов А.И.

**Москва 2022 г.**

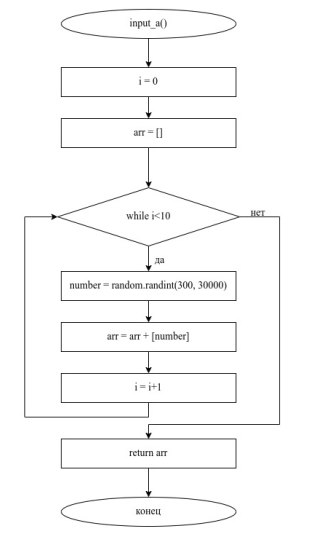
**1. Цель работы**

Изучение алгоритма сортировки слиянием.

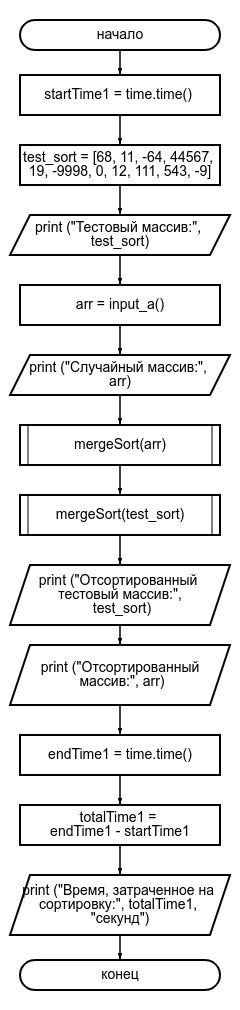
**2. Формулировка задачи**

Реализовать программу, выполняющую алгоритм сортировки слиянием.

**3. Блок-схема алгоритма**

****

****

****

Изображение выглядит как текст, часы

Автоматически созданное описание****



**4. Подбор тестовых примеров**

Ввод: массив из 10 элементов: 68, 11, -64, 44567, 19, -9998, 0, 12, 111, 543, -9.

Вывод: -9998, -64, 0, 11, 12, 19, 68, 111, 543, 44567.

**5. Код программы:**

import time

startTime1 = time.time()

def mergeSort(arr):

if len(arr)>1:

mid = len(arr)//2

lefthalf = arr[:mid]

righthalf = arr[mid:]

mergeSort(lefthalf)

mergeSort(righthalf)

i=0

j=0

k=0

while i<len(lefthalf) and j<len(righthalf):

if lefthalf[i]<righthalf[j]:

arr[k]=lefthalf[i]

i=i+1

else:

arr[k]=righthalf[j]

j=j+1

k=k+1

while i<len(lefthalf):

arr[k]=lefthalf[i]

i=i+1

k=k+1

while j<len(righthalf):

arr[k]=righthalf[j]

j=j+1

k=k+1

return arr

def input\_a():

import random

i = 0

arr = []

while i<100:

number = random.randint(300, 30000)

arr = arr + [number]

i = i+1

return arr

test\_sort = [68, 11, -64, 44567, 19, -9998, 0, 12, 111, 543, -9]

print ("Тестовый массив:", test\_sort)

arr = input\_a()

print ("Случайный массив:", arr)

mergeSort(arr)

mergeSort(test\_sort)

print ("Отсортированный тестовый массив:", test\_sort)

print ("Отсортированный массив:", arr)

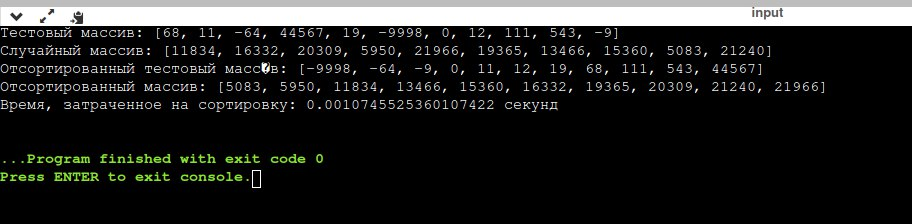
endTime1 = time.time() #время конца замера

totalTime1 = endTime1 - startTime1 #вычисляем затраченное время

print ("Время, затраченное на сортировку:", totalTime1, "секунд")

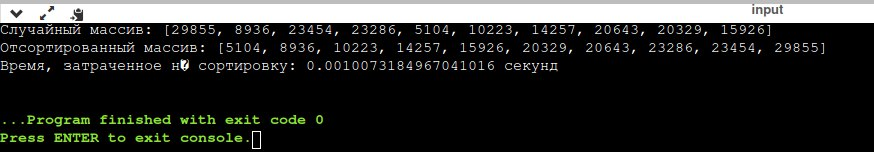
**6. Результат расчёта тестовых примеров на ПК**

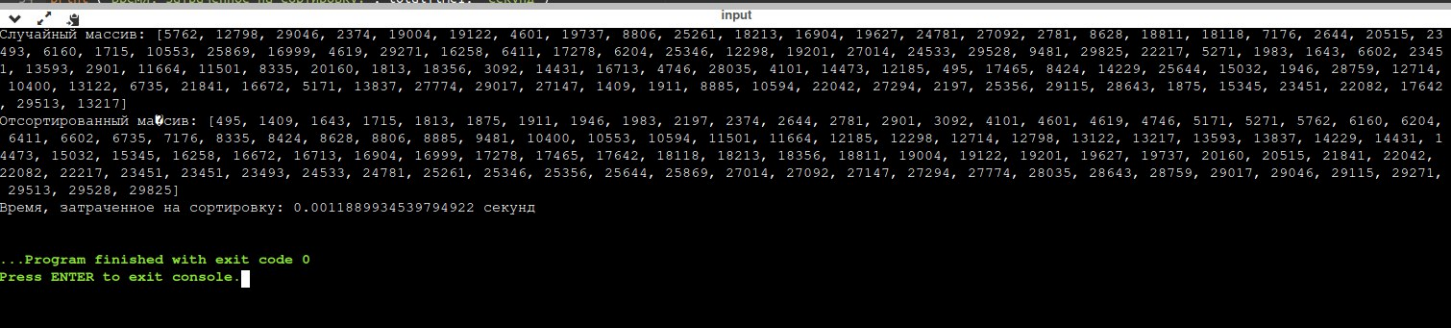
Проверим работоспособность алгоритма



Тестовый массив отсортирован верно.

Сравним время работы алгоритма. Для этого закомментируем обработку тестового массива.



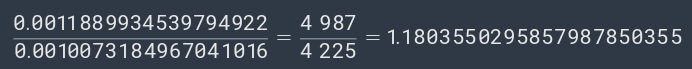


На сортировку массива из 10 элементов затрачено 0.0010073184967041016

секунд.

На сортировку массива из 100 элементов затрачено 0.00118899345397949222

секунд, то есть при увеличении количества элементов в 10 раз время увеличилось приблизительно в 1.18 раз.



**7. Вывод**

В данной работе я реализовала алгоритм сортировки слиянием на Python и протестировала его, а также узнала, что при увеличении количества элементов массива в 10 раз время сортировки слиянием увеличилось приблизительно в 1.18 раз.