**Федеральное агентство связи**

**Ордена Трудового Красного Знамени**

**Федеральное государственное бюджетное образовательное учреждение**

**Высшего образования**

**«Московский технический университет связи и информатики»**

Кафедра Математической кибернетики и информационных технологий

**Отчет по лабораторной работе №1**

По дисциплине «Структуры и алгоритмы обработки данных»

На тему:

«Методы сортировки»

Выполнил:

студент группы БВТ1904

Орлов Александр

Москва 2021

**Задание:**

Реализовать заданный метод сортировки строк числовой матрицы в соответствии с индивидуальным заданием. Для всех вариантов добавить реализацию быстрой сортировки (quicksort). Оценить время работы каждого алгоритма сортировки и сравнить его со временем стандартной функции сортировки, используемой в выбранном языке программирования.

9 вариант – Метод сортировки: обменом («метод пузырька»)

**Ход работы:**

Время работы в конце каждого кода указано без учета вывода матрицы на экран.

**Метод сортировки обменом**

*from random import randint*

*import time*

*t0 = time.process\_time()*

*"""*

*Method arrays*

*"""*

*def BildArray(n, m):*

*c = [[randint(100, 999) for j in range(m)] for i in range(n)]*

*d = [0]\*m*

*return c, d*

*"""*

*Method sort*

*"""*

*def bubble\_sort(arr):*

*n = len(arr)*

*swapped = True*

*x = -1*

*while swapped:*

*swapped = False*

*x = x + 1*

*for i in range(1, n-x):*

*if arr[i - 1] > arr[i]:*

*arr[i - 1], arr[i] = arr[i], arr[i - 1]*

*swapped = True*

*"""*

*Body program*

*"""*

*n, m = 500, 500*

*a, b = BildArray(n, m)*

*t = 0*

*while t != m:*

*for j in range(n):*

*b[j] = a[t][j]*

*bubble\_sort( b )*

*for j in range ( len(b) ):*

*print ( "{:4d}".format(b[j]), end = "" )*

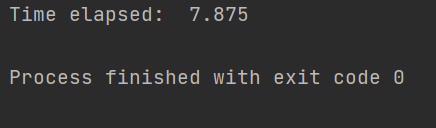
*print()*

*t += 1*

*t1 = time.process\_time() - t0*

*print("Time elapsed: ", t1 - t0)*

Время работы:



**Метод быстрой сортировки**

*from random import randint*

*import time*

*t0 = time.process\_time()*

*"""*

*Method arrays*

*"""*

*def BildArray(n, m):*

*c = [[randint(100, 999) for j in range(m)] for i in range(n)]*

*d = [0]\*m*

*return c, d*

*"""*

*Method sort*

*"""*

*def QuickSort(a, first, last):*

*if first >= last:*

*return*

*m = a[(last+first)//2]*

*b, e = first, last*

*while b <= e:*

*while a [b] < m:*

*b = b +1*

*while a [e] > m:*

*e = e -1*

*if b <= e :*

*a[b], a[e] = a [e], a [b]*

*b, e = b +1, e -1*

*QuickSort(a, first, e)*

*QuickSort(a, b, last)*

*"""*

*Body program*

*"""*

*n, m = 500, 500*

*a, b = BildArray(n, m)*

*t = 0*

*while t != m:*

*for j in range(n):*

*b[j] = a[t][j]*

*QuickSort(b, 0, len(b)-1)*

*for j in range ( len(b) ):*

*print ( "{:4d}".format(b[j]), end = "" )*

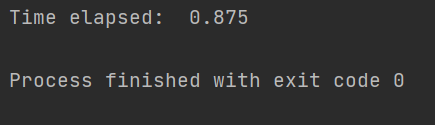
*print()*

*t += 1*

*t1 = time. process\_time () - t0*

*print("Time elapsed: ", t1 - t0)*

Время работы :



**Метод сортировки выбором**

*from random import randint*

*import time*

*t0 = time. process\_time ()*

*"""*

*Method arrays*

*"""*

*def BildArray(n, m):*

*c = [[randint(100, 999) for j in range(m)] for i in range(n)]*

*d = [0]\*m*

*return c, d*

*"""*

*Method sort*

*"""*

*def sel\_sort(array):*

*for i in range(len(array) - 1):*

*k = i*

*j = i + 1*

*while j < len(array):*

*if array[j] < array[k]:*

*k = j*

*j = j + 1*

*array[i], array[k] = array[k], array[i]*

*"""*

*Body program*

*"""*

*n, m = 500, 500*

*a, b = BildArray(n, m)*

*t = 0*

*while t != m:*

*for j in range(n):*

*b[j] = a[t][j]*

*sel\_sort( b )*

*for j in range ( len(b) ):*

*print ( "{:4d}".format(b[j]), end = "" )*

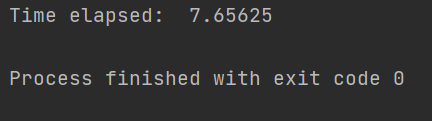
*print()*

*t += 1*

*t1 = time. process\_time () - t0*

*print("Time elapsed: ", t1 - t0)*

Время работы:



**Метод сортировки вставками**

*from random import randint*

*import time*

*t0 = time.process\_time()*

*"""*

*Method arrays*

*"""*

*def BildArray(n, m):*

*c = [[randint(100, 999) for j in range(m)] for i in range(n)]*

*d = [0]\*m*

*return c, d*

*"""*

*Method sort*

*"""*

*def insertion\_sort(array):*

*for i in range(1, len(array)):*

*temp = array[i]*

*j = i - 1*

*while (j >= 0 and temp < array[j]):*

*array[j + 1] = array[j]*

*j = j - 1*

*array[j + 1] = temp*

*"""*

*Body program*

*"""*

*n, m = 500, 500*

*a, b = BildArray(n, m)*

*t = 0*

*while t != m:*

*for j in range(n):*

*b[j] = a[t][j]*

*insertion\_sort( b )*

*for j in range ( len(b) ):*

*print ( "{:4d}".format(b[j]), end = "" )*

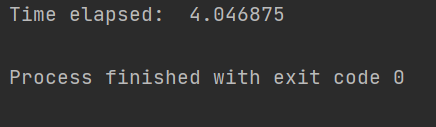
*print()*

*t += 1*

*t1 = time.process\_time() - t0*

*print("Time elapsed: ", t1 - t0)*

Время работы:



**Метод сортировки Шелла**

*from random import randint*

*import time*

*t0 = time.process\_time()*

*"""*

*Method arrays*

*"""*

*def BildArray(n, m):*

*c = [[randint(100, 999) for j in range(m)] for i in range(n)]*

*d = [0]\*m*

*return c, d*

*"""*

*Method sort*

*"""*

*def shell(array):*

*inc = len(array) // 2*

*while inc:*

*for i, el in enumerate(array):*

*while i >= inc and array[i - inc] > el:*

*array[i] = array[i - inc]*

*i -= inc*

*array[i] = el*

*inc = 1 if inc == 2 else int(inc \* 5.0 / 11)*

*"""*

*Body program*

*"""*

*n, m = 500, 500*

*a, b = BildArray(n, m)*

*t = 0*

*while t != m:*

*for j in range(n):*

*b[j] = a[t][j]*

*shell( b )*

*for j in range ( len(b) ):*

*print ( "{:4d}".format(b[j]), end = "" )*

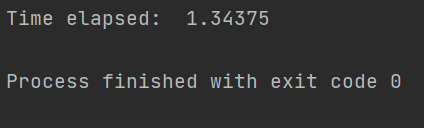
*print()*

*t += 1*

*t1 = time.process\_time() - t0*

*print("Time elapsed: ", t1 - t0)*

Время работы:



**Метод сортировки Пирамидальная(Турнирная)**

*from random import randint*

*import time*

*t0 = time.process\_time()*

*"""*

*Method arrays*

*"""*

*def BildArray(n, m):*

*c = [[randint(100, 999) for j in range(m)] for i in range(n)]*

*d = [0]\*m*

*return c, d*

*"""*

*Method sort*

*"""*

*def heapify(nums, heap\_size, root\_index):*

*largest = root\_index*

*left\_child = (2 \* root\_index) + 1*

*right\_child = (2 \* root\_index) + 2*

*if left\_child < heap\_size and nums[left\_child] > nums[largest]:*

*largest = left\_child*

*if right\_child < heap\_size and nums[right\_child] > nums[largest]:*

*largest = right\_child*

*if largest != root\_index:*

*nums[root\_index], nums[largest] = nums[largest], nums[root\_index]*

*heapify(nums, heap\_size, largest)*

*def heap\_sort(nums):*

*n = len(nums)*

*for i in range(n, -1, -1):*

*heapify(nums, n, i)*

*for i in range(n - 1, 0, -1):*

*nums[i], nums[0] = nums[0], nums[i]*

*heapify(nums, i, 0)*

*"""*

*Body program*

*"""*

*n, m = 500, 500*

*a, b = BildArray(n, m)*

*t = 0*

*while t != m:*

*for j in range(n):*

*b[j] = a[t][j]*

*heap\_sort(b)*

*for j in range ( len(b) ):*

*print ( "{:4d}".format(b[j]), end = "" )*

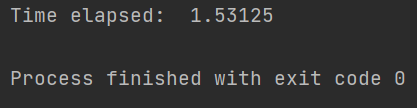
*print()*

*t += 1*

*t1 = time.process\_time() - t0*

*print("Time elapsed: ", t1 - t0)*

Время работы:



**Выводы:**

Было проведено сравнение видов сортировки с использованием языка программирования Python. Как показало исследование, быстрая сортировка дает наименьшие затраты по времени при сортировке матрицы.