Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №6
дисциплины «Алгоритмизация»

Выполнил:
Кожуховский Виктор Андреевич
2 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и
вычислительная техника»,
направленность (профиль)
«Программное обеспечение средств
вычислительной
техники и автоматизированных систем
», очная форма обучения

_____
(подпись)

Руководитель практики:
Воронкин Роман Александрович

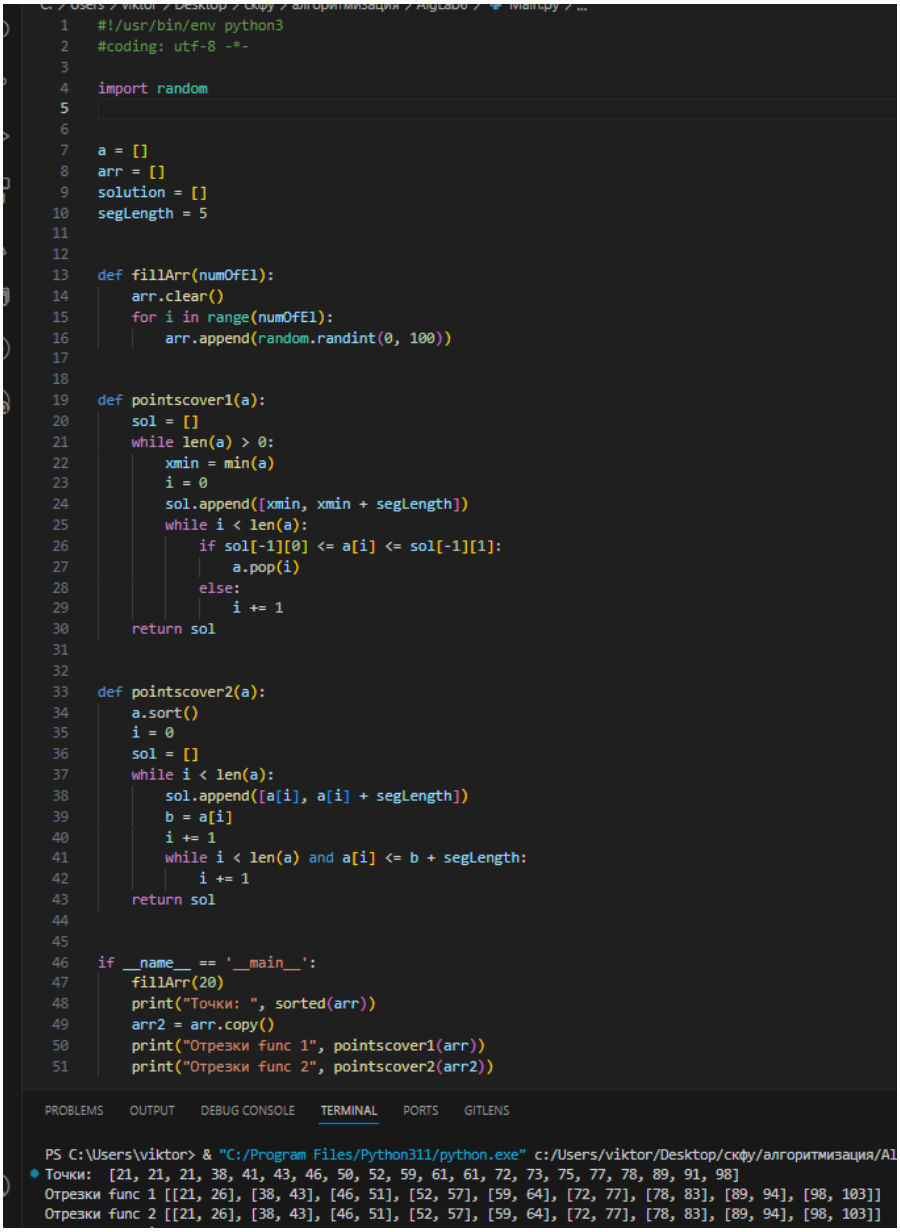
_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2023 г.

Порядок выполнения работы:

Написал программы для четырех жадных алгоритмов:



```python
#!/usr/bin/env python3
#coding: utf-8 -*-

import random


a = []
arr = []
solution = []
segLength = 5


def fillArr(numOfEl):
    arr.clear()
    for i in range(numOfEl):
        arr.append(random.randint(0, 100))


def pointscover1(a):
    sol = []
    while len(a) > 0:
        xmin = min(a)
        i = 0
        sol.append([xmin, xmin + segLength])
        while i < len(a):
            if sol[-1][0] <= a[i] <= sol[-1][1]:
                a.pop(i)
            else:
                i += 1
    return sol


def pointscover2(a):
    a.sort()
    i = 0
    sol = []
    while i < len(a):
        sol.append([a[i], a[i] + segLength])
        b = a[i]
        i += 1
        while i < len(a) and a[i] <= b + segLength:
            i += 1
    return sol


if __name__ == '__main__':
    fillArr(20)
    print("Точки: ", sorted(arr))
    arr2 = arr.copy()
    print("Отрезки func 1", pointscover1(arr))
    print("Отрезки func 2", pointscover2(arr2))
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  GITLENS

PS C:\Users\viktor> & "C:/Program Files/Python311/python.exe" c:/Users/viktor/Desktop/скфу/алгоритмизация/Al
Точки:  [21, 21, 21, 38, 41, 43, 46, 50, 52, 59, 61, 61, 72, 73, 75, 77, 78, 89, 91, 98]
Отрезки func 1 [[21, 26], [38, 43], [46, 51], [52, 57], [59, 64], [72, 77], [78, 83], [89, 94], [98, 103]]
Отрезки func 2 [[21, 26], [38, 43], [46, 51], [52, 57], [59, 64], [72, 77], [78, 83], [89, 94], [98, 103]]
```

Рисунок 1. Алгоритмы pointscover и pointscover улушенный

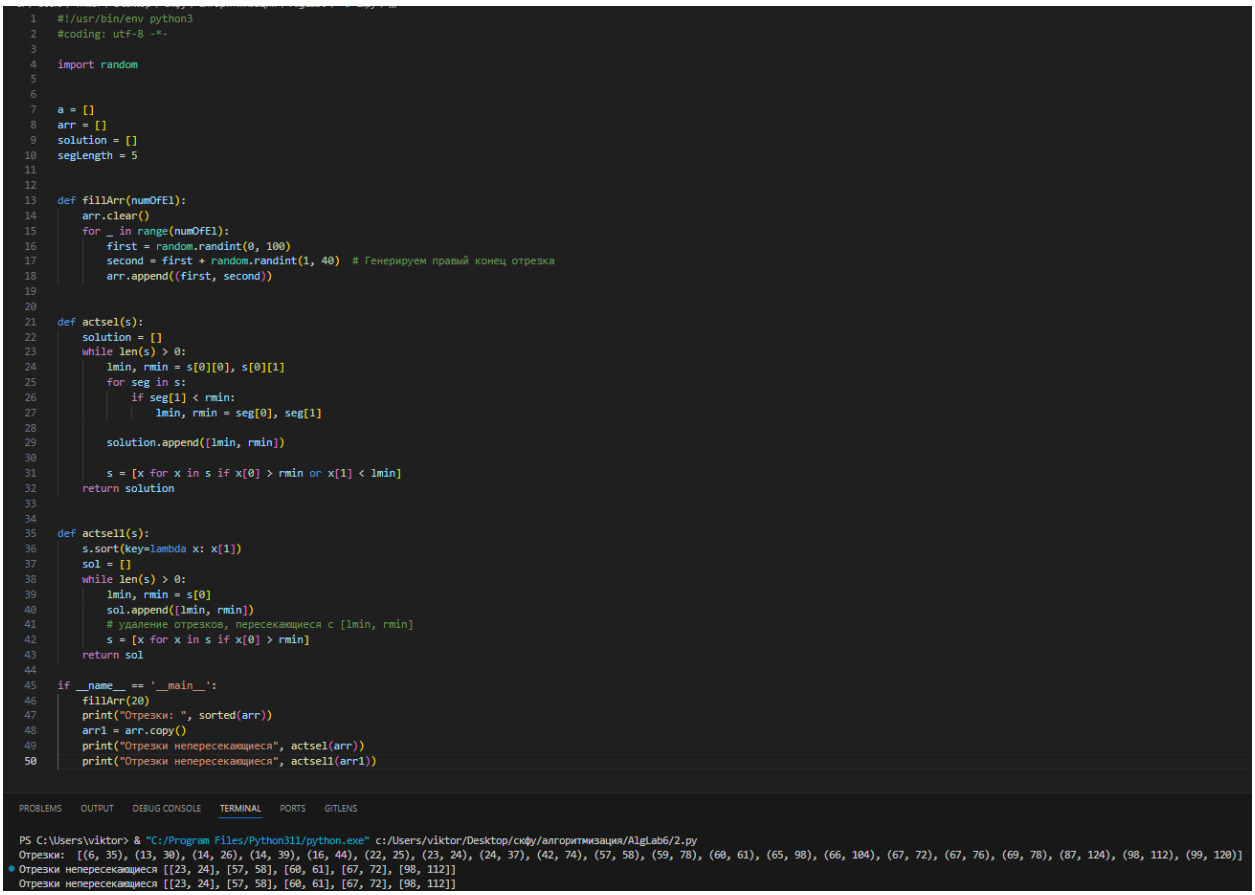


Рисунок 2. Алгоритмы ActSel и ActSel улушенный

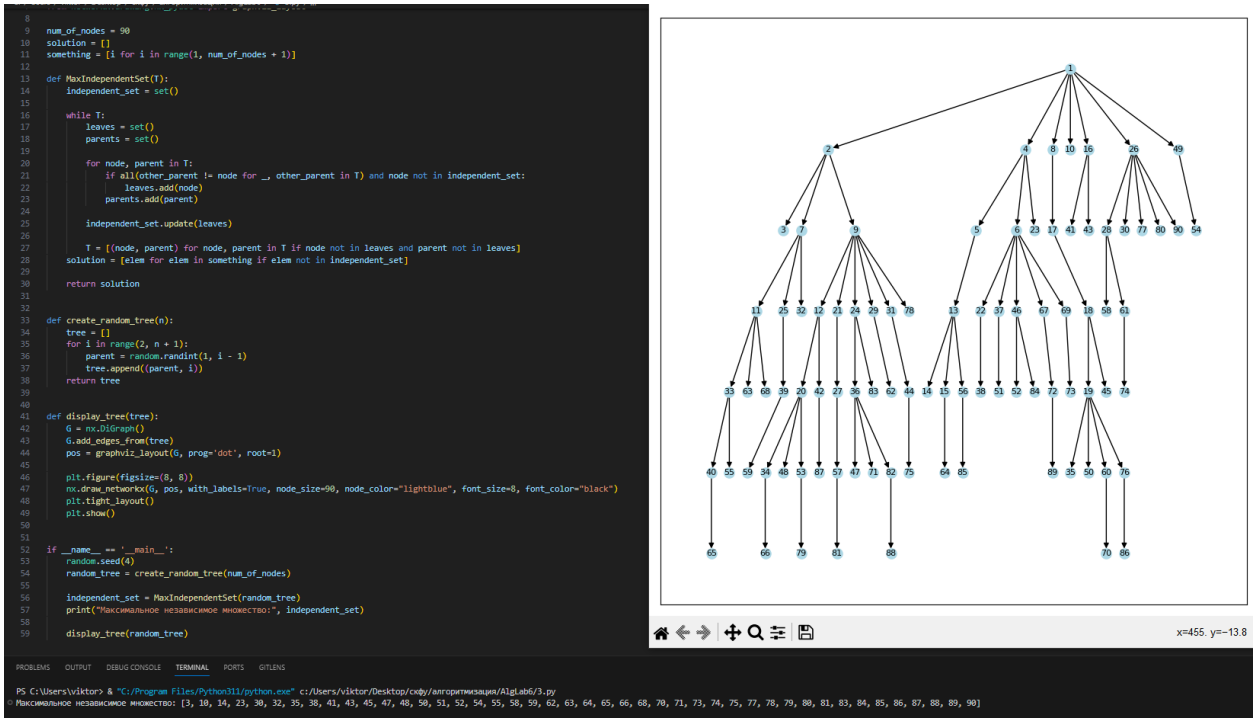
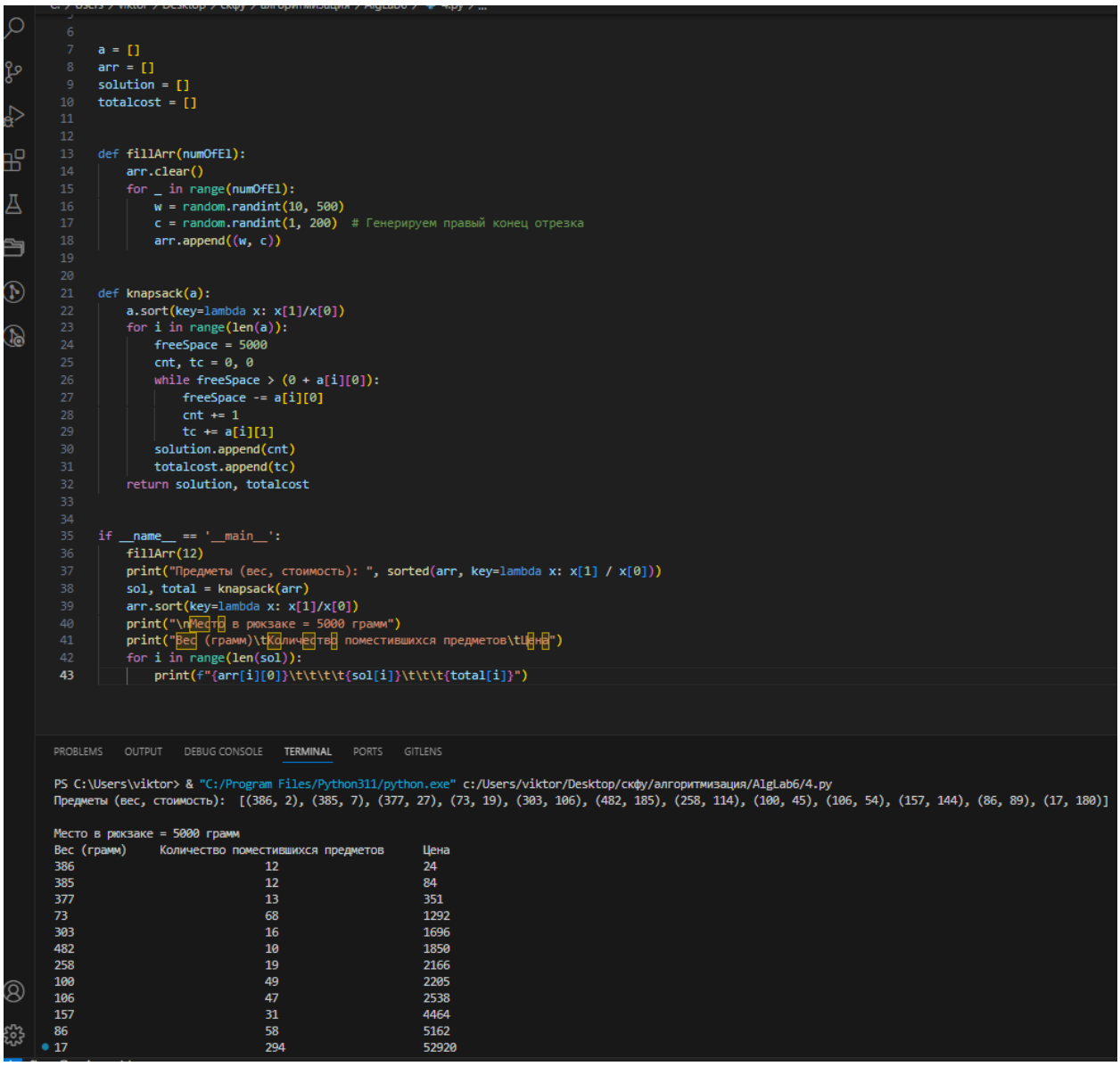


Рисунок 3. Алгоритм MaxIndependentSet



```python
a = []
arr = []
solution = []
totalcost = []


def fillArr(numOfEl):
    arr.clear()
    for _ in range(numOfEl):
        w = random.randint(10, 500)
        c = random.randint(1, 200)  # Генерируем правый конец отрезка
        arr.append((w, c))


def knapsack(a):
    a.sort(key=lambda x: x[1]/x[0])
    for i in range(len(a)):
        freeSpace = 5000
        cnt, tc = 0, 0
        while freeSpace > (0 + a[i][0]):
            freeSpace -= a[i][0]
            cnt += 1
            tc += a[i][1]
        solution.append(cnt)
        totalcost.append(tc)
    return solution, totalcost


if __name__ == '__main__':
    fillArr(12)
    print("Предметы (вес, стоимость): ", sorted(arr, key=lambda x: x[1] / x[0]))
    sol, total = knapsack(arr)
    arr.sort(key=lambda x: x[1]/x[0])
    print("\nМесто в рюкзаке = 5000 грамм")
    print("Вес (грамм)\tКоличество поместившихся предметов\tЦена")
    for i in range(len(sol)):
        print(f"{arr[i][0]}\t\t\t\t{sol[i]}\t\t\t{total[i]}")
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  GITLENS

PS C:\Users\viktor> & "C:/Program Files/Python311/python.exe" c:/Users/viktor/Desktop/скфу/алгоритмизация/AlgLab6/4.py
Предметы (вес, стоимость):  [(386, 2), (385, 7), (377, 27), (73, 19), (303, 106), (482, 185), (258, 114), (100, 45), (106, 54), (157, 144), (86, 89), (17, 180)]

Место в рюкзаке = 5000 грамм
Вес (грамм)     Количество поместившихся предметов      Цена
386                             12                      24
385                             12                      84
377                             13                      351
73                              68                      1292
303                             16                      1696
482                             10                      1850
258                             19                      2166
100                             49                      2205
106                             47                      2538
157                             31                      4464
86                              58                      5162
17                              294                     52920
```

Рисунок 4. Алгоритм KnapSack

Вывод: в результате выполнения лабораторной работы были изучены четыре жадных алгоритма.