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»
Вариант 5.

Выполнила:
Михеева Елена Александровна
2 курс, группа ИВТ-б-з-20-1,
09.03.01 «Информатика и
вычислительная техника»,
направленность (профиль)
«Программное обеспечение средств
вычислительной
техники и автоматизированных
систем», очная форма обучения

_____
(подпись)

Руководитель практики:
Воронкин Р.А...,

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2023 г.

Порядок выполнения работы.

1. Были написана программа, которая на вход принимает множество точек, а выводит минимальное количество отрезков единичной длины, которые покрывают все точки.

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

# Покрытие точек отрезками
# Наивный алгоритм

def PointsCover(S):
    result = []
    while len(S) > 0:
        Xm = min(S)
        result.append([Xm, Xm + 1])
        while Xm in S:
            S.remove(Xm)
        while Xm+1 in S:
            S.remove(Xm+1)
    return result


if __name__ == '__main__':
    S = [3, 8, 2, 9, 1, 2, 5, 5, 6]
    print(PointsCover(S))
```

```
ПРОБЛЕМЫ  ТЕРМИНАЛ  ···  Python Debug Console

(base) mikheeva@MacBook-Pro-Elena Algorithm_6 % python3 Ex_1.1.py
[[1, 2], [3, 4], [5, 6], [8, 9]]
```

Рисунок 1. Покрытие точек отрезками (наивный алгоритм)

2. Написана программа для покрытия точек единичными отрезками по улучшенному алгоритму.

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

# Покрытие точек отрезками
# Улучшенный алгоритм


def PointsCover(S):
    S.sort()
    result = []
    i = 0
    while i < len(S):
        Xm = S[i]
        result.append([S[i], S[i] + 1])
        i += 1
        while i < len(S) and S[i] <= Xm + 1:
            i += 1
    return result


if __name__ == '__main__':
    S = [3, 8, 2, 9, 1, 2, 5, 5, 6]
    print(PointsCover(S))
```

```
ПРОБЛЕМЫ  ТЕРМИНАЛ  ···  Python Debug Console

(base) mikheeva@MacBook-Pro-Elena Algorithm_6 % python3 Ex_1.2.py
[[1, 2], [3, 4], [5, 6], [8, 9]]
```

Рисунок 2. Покрытие точек отрезками (улучшенный алгоритм)

3. Была написана программа, которая на вход получает множество отрезков на прямой, а выводит максимальное количество попарно не пересекающихся отрезков.

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

# Задача о выборе заявок
# Наивный алгоритм


def ActSel(S):
    result = []
    while len(S) != 0:
        min_right_end = min(S, key=lambda x: x[1])
        result.append(min_right_end)
        S = [segment for segment in S if segment[0] > min_right_end[1]]

    return result


if __name__ == '__main__':
    S = [(2, 3), (2, 4), (3, 5), (4, 6), (1, 2)]
    print(ActSel(S))
```

```
ПРОБЛЕМЫ  ТЕРМИНАЛ  ···  Python Debug Console

(base) mikheeva@MacBook-Pro-Elena Algorithm_6 % python3 Ex_2.1.py
[(1, 2), (3, 5)]
```

Рисунок 3. Задача о выборе заявок (наивный алгоритм)

4. Написали программу для «выбора заявок» по улучшенному алгоритму.

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

# Задача о выборе заявок
# Улучшенный алгоритм


def ActSel(S):
    S.sort(key=lambda x: x[1])
    result = []
    last_end = float('-inf')
    for segment in S:
        if segment[0] > last_end:
            result.append(segment)
            last_end = segment[1]
    return result


if __name__ == '__main__':
    S = [(2, 3), (2, 4), (3, 5), (4, 6), (1, 2)]
    print(ActSel(S))
```

```
ПРОБЛЕМЫ  ТЕРМИНАЛ  ···  Python Debug Console

python3: error: SDK "macosx" cannot be located
[(1, 2), (3, 5)]
```

Рисунок 4. Задача о выборе заявок (улучшенный алгоритм)

5. Была реализована программа, которая на вход получает дерево, а на выходе максимальное независимое множество вершин. Для этого была написана функция maxIndependentSet(), которая пока граф не пустой делит его на 2 множества вершин: родителей и детей. С помощью разности этих множеств ищет листья графа. Затем удаляет из графа листья и их родителей.

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

# Планирование вечеринки в компании


def delete(T, leave, parents):
    result = []
    fordelete = leave | parents

    for segment in T:
        if segment[1] not in fordelete:
            result.append(segment)

    return result


def maxIndependentSet(T):
    result = set()
    vertices = set(range(1, len(T)+1))

    while len(T) != 0:
        # Определила 2 множества с родителями и детьми
        par = [segment[0] for segment in T]
        child = [segment[1] for segment in T]

        # Множество листьев
        leave = set(child).difference(set(par))
        result = result | leave

        # Определила родителей для листьев
        leave_parents = set()
        for segment in T:
            if segment[1] in leave:
                leave_parents.add(segment[0])

        # Из множества вершин удалила листья и их родителей
        vertices -= (leave | leave_parents)
        T = delete(T, leave, leave_parents)

        if len(vertices) == 1 and len(T) == 0:
            result |= vertices

    return result


if __name__ == '__main__':
    graf = ([1, 2], [1, 3], [1, 4], [2, 5], [3, 6],
            [4, 7], [4, 8], [7, 9], [7, 10], [7, 11])

    print("Ответ: ", maxIndependentSet(graf))

```

ПРОБЛЕМЫ ВЫХОДНЫЕ ДАННЫЕ КОНСОЛЬ ОТЛАДКИ ТЕРМИНАЛ ПОРТЫ Python Debug Console

```
(base) mikheeva@MacBook-Pro-Elena Algorithm_6 % python3 Ex_3.py
Ответ:  {1, 5, 6, 8, 9, 10, 11}
```

Рисунок 5. Задача «Планирование вечеринки в компании»

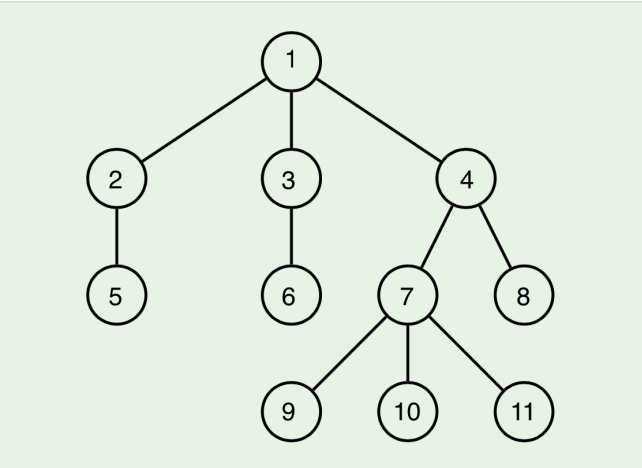

Рисунок 6. Исходный граф

6. Написали программу, которая на вход получает двумерный массив с весом и ценой предметов и вместимость рюкзака. Функция Knapsack() ищет суммарную стоимость частей предметов суммарного веса не более веса рюкзака.

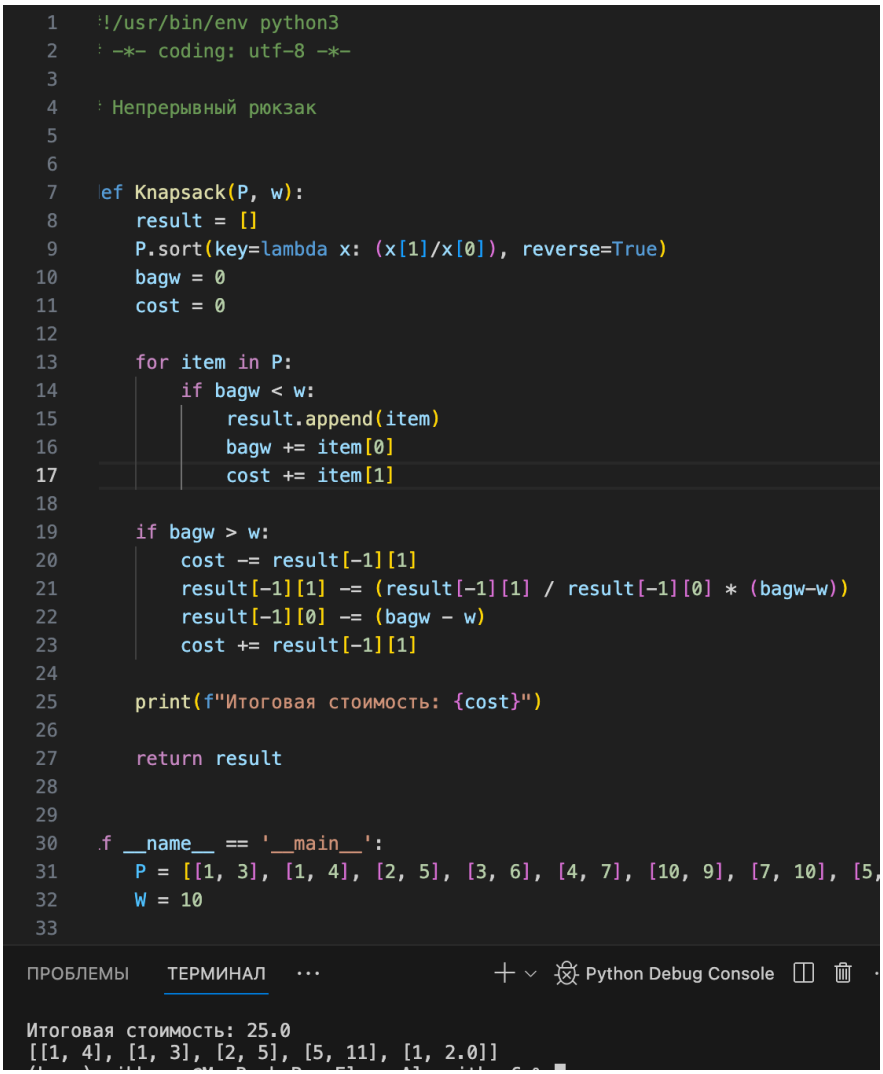

Рисунок 7. Задача «непрерывный рюкзак»