Министерство науки и высшего образования

Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Рыбинский государственный авиационный технический университет имени П. А. Соловьева»

Институт информационных технологий и систем управления

Кафедра математического и программного обеспечения электронных вычислительных средств

ЛАБОРАТОРНАЯ РАБОТА №6

по дисциплине:
«Исследование операций»

Отчет

Студент группы ИПБ-22_____Ушаков М. С.

Преподаватель _____Задорина Н. А.

Рыбинск 2024

1. Формулировка задачи

Составить математическую модель задачи и схему вычислений. Разработать программу, реализующую метод. Получить решение задачи. Проверить правильность в Excel или MathCad

Депутат округа принимает участие в переизбрании на следующий срок. Денежные средства на предвыборную кампанию составляют примерно 10000 уе. Хотя комитет по переизбранию хотел бы провести кампанию в пяти избирательных участках округа, ограниченность денежных средств предписывает действовать по-другому, Приведенная ниже таблица содержит данные о числе избирателей и денежных средствах, необходимых для проведения успешной кампании по каждому избирательному участку. Каждый участок может либо использовать все предназначенные деньги, либо вовсе их не использовать. Как следует распределить денежные средства?

Участок	Число избирателей	Необходимые денежные средства
1	3100	3500
2	2600	2500
3	3500	4000
4	2800	3000
5	2400	2000

2. Результаты расчетов в Excel

	A	B	C	D	E	F	G
1	x1	x2	x3	x4	x5	Число ибирателей	По средствам
2	1	1	1	1	1	14400	ЛОЖЬ
3	1	1	1	1	0	12000	ЛОЖЬ
4	1	0	1	1	1	11800	ЛОЖЬ
5	1	1	1	0	1	11600	ЛОЖЬ
6	0	1	1	1	1	11300	ЛОЖЬ
7	1	1	0	1	1	10900	ЛОЖЬ
8	1	0	1	1	0	9400	ЛОЖЬ
9	1	1	1	0	0	9200	ИСТИНА
10	1	0	1	0	1	9000	ИСТИНА
11	0	1	1	1	0	8900	ИСТИНА
12	0	0	1	1	1	8700	ИСТИНА
13	0	1	1	0	1	8500	ИСТИНА
14	1	1	0	1	0	8500	ИСТИНА
15	1	0	0	1	1	8300	ИСТИНА
16	1	1	0	0	1	8100	ИСТИНА
17	0	1	0	1	1	7800	ИСТИНА
18	1	0	1	0	0	6600	ИСТИНА
19	0	0	1	1	0	6300	ИСТИНА
20	0	1	1	0	0	6100	ИСТИНА
21	0	0	1	0	1	5900	ИСТИНА
22	1	0	0	1	0	5900	ИСТИНА
23	1	1	0	0	0	5700	ИСТИНА
24	1	0	0	0	1	5500	ИСТИНА
25	0	1	0	1	0	5400	ИСТИНА
26	0	0	0	1	1	5200	ИСТИНА
27	0	1	0	0	1	5000	ИСТИНА
28	0	0	1	0	0	3500	ИСТИНА
29	1	0	0	0	0	3100	ИСТИНА
30	0	0	0	1	0	2800	ИСТИНА
31	0	1	0	0	0	2600	ИСТИНА
32	0	0	0	0	1	2400	ИСТИНА
33	0	0	0	0	0	0	ИСТИНА

3. Результаты расчета программы

```
План: 11100
Сумма избирателей: 9200
Необходимые средства: 10000
```

Код программы см. в Приложении А

4. Вывод

В результате работы мы смогли найти оптимальное распределение денежных средств для получения максимального охвата избирателей. Была написана программа, которая рассчитала, что максимальная прибыль может быть получена если выделить деньги на первый, второй и третий участки.

Полученное решение было проверено при помощи Excel. Совпадение ответов подтверждает корректность найденного решения.

Приложение А.

```
w = {0: 3500, 1: 2500, 2: 4000, 3: 3000, 4: 2000}
n = {0: 3100, 1: 2600, 2: 3500, 3: 2800, 4: 2400}

class Tree:

    def __init__(self, w, n, path, left=None, right=None):
        self.path = path
        self.cur_w = w
        self.cur_n = n
        self.left = left
        self.right = right

    def __str__(self):
        rightStr = str(self.right) if self.right is not None else ""
        tabbedRightStr = "\t\t" + rightStr.replace("\n", "\n\t\t")

        leftStr = str(self.left) if self.left is not None else ""
        tabbedLeftStr = "\t\t" + leftStr.replace("\n", "\n\t\t")

        return "\n".join(
            filter(
                lambda x: x.strip(),
                (
                    tabbedRightStr,
                    (
                      f"План: {self.path}\nСумма избирателей: {self.cur_n}\nНеобходимые средства:
{abs(self.cur_w-10000)}"
                    ),
                    tabbedLeftStr,
                ),
            )
        )

tree = Tree(10000, 0, "")
points = [tree]
for i in range(count):
    new_points = []
    for point in points:
        point.left = Tree(point.cur_w, point.cur_n, point.path + "0")
        new_points.append(point.left)
        if point.cur_w - w[i] >= 0:
            point.right = Tree(point.cur_w - w[i], point.cur_n + n[i], point.path + "1")
            new_points.append(point.right)
    points = new_points

print(max(points, key=lambda points: points.cur_n))
```