



Отчёт по выполнению практического задания № 8.2
Тема:
**«Реализация алгоритмов на основе сокращения числа
переборов»**

Группа: ИКБО-10-23

Москва 2024

Вариант:

Номер: 6

Задача:

Дано прямоугольное поле размером $n \times m$ клеток.

Можно совершать шаги длиной в одну клетку вправо, вниз или по диагонали вправо-вниз. В каждой клетке записано некоторое натуральное число. Необходимо попасть из верхней левой клетки в правую нижнюю. Вес маршрута – это сумма чисел всех посещенных клеток. Найти маршрут с минимальным весом.

Метод:

Динамическое программирование.

1.1 Формулировка задачи

1. Разработать алгоритм решения задачи с применением метода, указанного в варианте и реализовать программу.
2. Оценить количество переборов при решении задачи стратегией «в лоб» - грубой силы. Сравнить с числом переборов при применении метода.
3. Оформить отчет в соответствии с требованиями документирования разработки ПО.

1.2 Ход решения

Для переборного решения необходимо проверить все возможные пути из левой верхней клетки в правую нижнюю, чьё число возрастает очень быстро.

Для эффективного решения достаточно лишь каждый раз для каждой клетки выбирать из трех возможных чисел, откуда можно в неё попасть, наименьшее и прибавлять к нему собственное значение клетки.

Таким образом после применения алгоритма ко всем клеткам в после завершения работы алгоритма получим в правой нижней клетке минимальное возможное число.

1.3 Исходный код программ

На рисунках 1-2 представлены исходный коды функций нахождения количества возможных вариантов и нахождения минимальной суммы.

```
8  vector<vector<int>> count_variants(int n, int m)
9  {
10     vector<vector<int>> arr(n, vector<int>(m, 0));
11     arr[0][0] = 1;
12     for (int i = 1; i < n; ++i)
13     {
14         arr[i][0] += arr[i - 1][0];
15     }
16     for (int j = 1; j < m; ++j)
17     {
18         arr[0][j] += arr[0][j - 1];
19     }
20     for (int i = 1; i < n; ++i)
21     {
22         for (int j = 1; j < m; ++j)
23         {
24             arr[i][j] += arr[i - 1][j] + arr[i][j - 1] + arr[i - 1][j - 1];
25         }
26     }
27     return arr;
28 }
```

Рисунок 1 — Функция нахождения количества путей

```

43 vector<vector<int>> min_sum(vector<vector<int>> &base_arr)
44 {
45     int n = base_arr.size();
46     int m = base_arr[0].size();
47     vector<vector<int>> arr(n, vector<int>(m, 0));
48     arr[0][0] = base_arr[0][0];
49     for (int i = 1; i < n; ++i)
50     {
51         arr[i][0] = arr[i - 1][0] + base_arr[i][0];
52     }
53     for (int j = 1; j < m; ++j)
54     {
55         arr[0][j] = arr[0][j - 1] + base_arr[0][j];
56     }
57     for (int i = 1; i < n; ++i)
58     {
59         for (int j = 1; j < m; ++j)
60         {
61             arr[i][j] = base_arr[i][j] + min(arr[i - 1][j], arr[i][j - 1], arr[i - 1][j - 1]);
62         }
63     }
64     return arr;
65 }

```

Рисунок 2 — Функция нахождения минимальной суммы

1.4 Результаты тестирования

На рисунках 3-4 представлены результаты тестирования программы.

Исходная матрица:							
924	943	242	825	805	802	561	47
376	971	477	246	698	50	869	605
233	847	321	504	426	375	398	161
270	326	945	620	208	928	964	624
209	869	162	133	364	547	92	650
158	35	52	605	965	273	934	63
Матрица количества вариантов:							
1	1	1	1	1	1	1	1
1	3	5	7	9	11	13	15
1	5	13	25	41	61	85	113
1	7	25	63	129	231	377	575
1	9	41	129	321	681	1289	2241
1	11	61	231	681	1683	3653	7183
Матрица минимальных значений:							
924	1867	2109	2934	3739	4541	5102	5149
1300	1895	2344	2355	3053	3103	3972	4577
1533	2147	2216	2720	2781	3156	3501	3662
1803	1859	2804	2836	2928	3709	4120	4125
2012	2672	2021	2154	2518	3065	3157	3807
2170	2047	2073	2626	3119	2791	3725	3220

Рисунок 7 — Результаты тестирования программы

Исходная матрица:													
547	656	623	604	467	69	892	42	522	448	814	918	100	924
573	39	338	452	306	615	593	122	225	962	208	717	35	597
483	337	637	166	527	244	629	862	155	644	802	804	522	723
517	447	728	538	631	905	418	960	340	730	263	582	764	195
296	824	59	713	472	937	801	41	1	409	284	126	56	389
231	68	668	876	143	297	651	958	580	340	102	664	875	271
790	914	240	703	284	465	736	806	170	743	267	296	128	26
646	434	709	574	981	55	56	48	764	218	398	847	427	168
683	502	51	575	184	189	295	244	12	807	295	457	675	473
50	232	509	981	992	530	947	396	816	570	720	34	785	412
Матрица количества вариантов:													
1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	3	5	7	9	11	13	15	17	19	21	23	25	27
1	5	13	25	41	61	85	113	145	181	221	265	313	365
1	7	25	63	129	231	377	575	833	1159	1561	2047	2625	3303
1	9	41	129	321	681	1289	2241	3649	5641	8361	11969	16641	22569
1	11	61	231	681	1683	3653	7183	13073	22363	36365	56695	85305	124515
1	13	85	377	1289	3653	8989	19825	40081	75517	134245	227305	369305	579125
1	15	113	575	2241	7183	19825	48639	108545	224143	433905	795455	1392065	2340495
1	17	145	833	3649	13073	40081	108545	265729	598417	1256465	2485825	4673345	8405905
1	19	181	1159	5641	22363	75517	224143	598417	1462563	3317445	7059735	14218905	27298155
Матрица минимальных значений:													
547	1203	1826	2430	2897	2966	3858	3900	4422	4870	5684	6602	6702	7626
1120	586	924	1376	1682	2297	2890	3012	3237	4199	4407	5124	5159	5756
1603	923	1223	1090	1617	1861	2490	3352	3167	3811	4613	5211	5646	5882
2120	1370	1651	1628	1721	2522	2279	3239	3507	3897	4074	4656	5420	5615
2416	2194	1429	2142	2100	2658	3080	2320	2321	2730	3014	3140	3196	3585
2647	2262	2097	2305	2243	2397	3048	3278	2900	2661	2763	3427	4015	3467
3437	3176	2337	2800	2527	2708	3133	3854	3070	3404	2928	3059	3187	3213
4083	3610	3046	2911	3508	2582	2638	2686	3450	3288	3326	3775	3486	3355
4766	4112	3097	3486	3095	2771	2877	2882	2698	3505	3583	3783	4161	3828
4816	4344	3606	4078	4087	3301	3718	3273	3514	3268	3988	3617	4402	4240

Рисунок 8 — Результаты тестирования программы

2.1 Выводы

Для большинства задач помимо полного переборного решения существуют также и эффективные алгоритмы, позволяющие во много раз сократить сложность задачи и время решения.