实验报告

课程名称	算法设计与分析	班级	21 杨班
实验编号	二	姓名	皮昊旋
实验名称	Dp 和贪心	学号	2021428010127

题目 1： 01 背包

➢ 思路描述

① 将所需数据存进相应的变量和数组，同时申请一个 dp 二维数组，用于存放动态规划的过程
② 遍历循环 dp 数组，其实就是便利循环给出的物品和背包剩余空间。
③ 如果背包空间不够，那么直接丢弃不要，即 dp[p][q] = dp[p - 1][q];
如果空间还够，那么久判断是否要拿 dp[p][q] = max(dp[p - 1][q], dp[p - 1][q - weight[p]] + value[p]);
④ 同时在判断时，用 path 二维数组记录拿物品的过程
⑤ 输出 dp 数组右下角的值就是最大价值
⑥ 筛选输出 path 数组最后一行的数字，就是已拿物品的编号

➢ 代码

```
➢  #include <bits/stdc++.h>
➢  using namespace std;
➢
➢  int n = 0; // number of goods
➢  int weight[11] = {0};
➢  int value[11] = {0};
➢  int dp[111][111] = {{0}};
➢  int path[111][111] = {{0}};
➢  int space = 10;
➢  int p = 0, q = space;
➢
➢  int main(int argc, char const *argv[])
➢  {
➢      cout << "input the number of goods( NOT bigger than 7):"
   << endl;
➢      cin >> n;
```

```
➢       cout << "input every good's weight in next line using
    space to split it :" << endl;
➢       for (int i = 1; i <= n; i++)
➢       {
➢           cin >> weight[i];
➢       }
➢       cout << "input every good's value in next line using space
    to split it :" << endl;
➢       for (int i = 1; i <= n; i++)
➢       {
➢           cin >> value[i];
➢       }
➢       for (p = 1; p <= n; p++)
➢       {
➢           for (q = 1; q <= space; q++)
➢           {
➢               if (weight[p] > q)
➢               {
➢                   dp[p][q] = dp[p - 1][q];
➢                   path[p][q] = path[p - 1][q];
➢               }
➢               else
➢               {
➢                   dp[p][q] = max(dp[p - 1][q], dp[p - 1][q -
    weight[p]] + value[p]);
➢                   if (dp[p][q] == dp[p - 1][q - weight[p]] +
    value[p])
➢                   {
➢                       path[p][q] = p;
➢                   }
➢                   else
➢                   {
➢                       path[p][q] = path[p - 1][q];
➢                   }
➢               }
➢           }
➢       }
➢       cout << endl;
➢       cout << dp[n][space] << endl;
➢       for (int z = 1; z <= space; z++)
➢       {
➢           if (path[n][z] > path[n][z - 1])
➢           {
➢
```

```
            cout << path[n][z] << " ";
        }
    }
    cout << endl;
    return 0;
}
```

- 运行截图

```
root@pihaoxuan--v:/home/pihaoxuan/algorithm# ./01-bag-dp
input the number of goods( NOT bigger than 7):
5
input every good's weight in next line using space to split it :
2 2 6 5 4
input every good's value in next line using space to split it :
6 3 5 4 6

15
1 2 5
root@pihaoxuan--v:/home/pihaoxuan/algorithm#
```

题目 2： 题目名称...

思路：

① 创建需要的变量存储数据，同时创建一个变量 remain 存储剩余需要取的货币的价值；

② 对货币面值的数组排序，从小到大，之后处理的时候就从数组末尾开始拿，确保是贪心

③ While 循环，进行兑换计算，如果剩余需要取的货币价值小于当前货币，那么不能取，否则先计算要取当前货币多少张，才可以不取，算完之后减去面值×数量

④ 输出张数

代码：

```
#include <bits/stdc++.h>
using namespace std;

int n = 0;
int value[105] = {0};
int remain = 0;
int test = -1;

int p = 0;
```

```
int main(int argc, char const *argv[])
{
    int count = 0;
    cin >> n;
    for (int i = 1; i <= n; i++)
    {
        cin >> value[i];
    }
    sort(value, value + n);
    while (test != 0)
    {
        cin >> test;
        remain = test;
        count = 0; // 将计数器归零
        for (int i = n; i >= 1; --i)
        {
            if (remain < value[i])
                continue; // 使用 continue 跳过本次循环
            else
            {
                count += floor(remain / value[i]);
                remain = remain - floor(remain / value[i]) *
value[i]; // 使用 floor 取整
            }
        }
        cout << "RMB  " << test << "  need  " << count << endl;
    }
    return 0;
}
```

运行结果：

```
root@pihaoxuan--v:/home/pihaoxuan/algorithm# ./pay
4 1 2 5 10
8
RMB  8  need  3
11
RMB  11  need  2
43
RMB  43  need  6
167
RMB  167  need  18
0
RMB  0  need  0
root@pihaoxuan--v:/home/pihaoxuan/algorithm#
```

题目 3： 题目名称...

思路：

① 创建变量存储数据，同时创建一个 path 一维数组存储加油的位点和一个 remain 变量存储当前状态汽车还剩多少油。

② 输入读取数据

③ 如果当前 remain 不够下一个加油站的数值了，必须加油，remain 等于 n，path 记录当前站点编号，count 加一

④ 否则的话，remain 减去下一个站点距离现在的距离

⑤ 输出 count 和 path 数组中的非零值。

代码：

```cpp
#include <bits/stdc++.h>
using namespace std;

int n = 0;         // total distance
int m = 0;         // number of gas station
int d[1024] = {0}; // distance array
int path[1024] = {0};
int p = 0; // pointer of path
int remain = 0;

int main(int argc, char const *argv[])
{
    int count = 0;
    cin >> n;
    cin >> m;
    for (int i = 1; i <= m; i++)
    {
        cin >> d[i];
```

```
    }

    remain = n;
    for (int j = 1; j <= m; j++)
    {
        if (remain < d[j])
        {
            remain = n; // full the gas
            path[p++] = j;
            count++;
        }
        else
        {
            remain -= d[j];
        }
    }

    cout << count << endl;
    for (int i = 0; i < m + 1; i++)
    {
        if (path[i] != 0)
        {
            cout << path[i] << " ";
        }
    }
    cout << endl;

    return 0;
}
```

运行截图：