中山大学计算机学院

算法设计与分析

本科生实验报告

（2023学年秋季学期）
课程名称：Algorithm design and analysis

教学班级	专业(方向)	学号	姓名
计科二班	计算机科学与技术	21307185	张礼贤

算法原理

对于页面的分配问题，可以将其抽象为在一定范围内的页面二分查找问题，并对页面数量的范围确定进行特化，下面为分点描述：

- **基本思路：**
  - 将问题抽象为桶分组问题，每个学生作为一个**桶**，每本书表示**物品**，每本书的页数表示**物品大小**，桶的容量决定能容纳的大小，即每个学生阅读的书页数目。这样问题就变成了找到一个**容量最小的桶**，使得他的容量可以在满足学生个数的同时容纳所有的物品。
- **搜索策略：**
  - 采用二分搜索的方式，初始搜索范围为[max(array), sum(array)],即从数组中最大的单个值到数组中所有数据的和。一开始定义mid = (start + end) / 2 作为初始搜索容量
    1. 如果对于容量 mid 发现偏小，则搜索空间变成[start, mid - 1]
    2. 如果对于容量 mid 发现偏大，则搜索空间变成[mid + 1, end]
    3. 如果对于容量 mid 刚好合适，则找到答案，返回解
- **容量判定：**

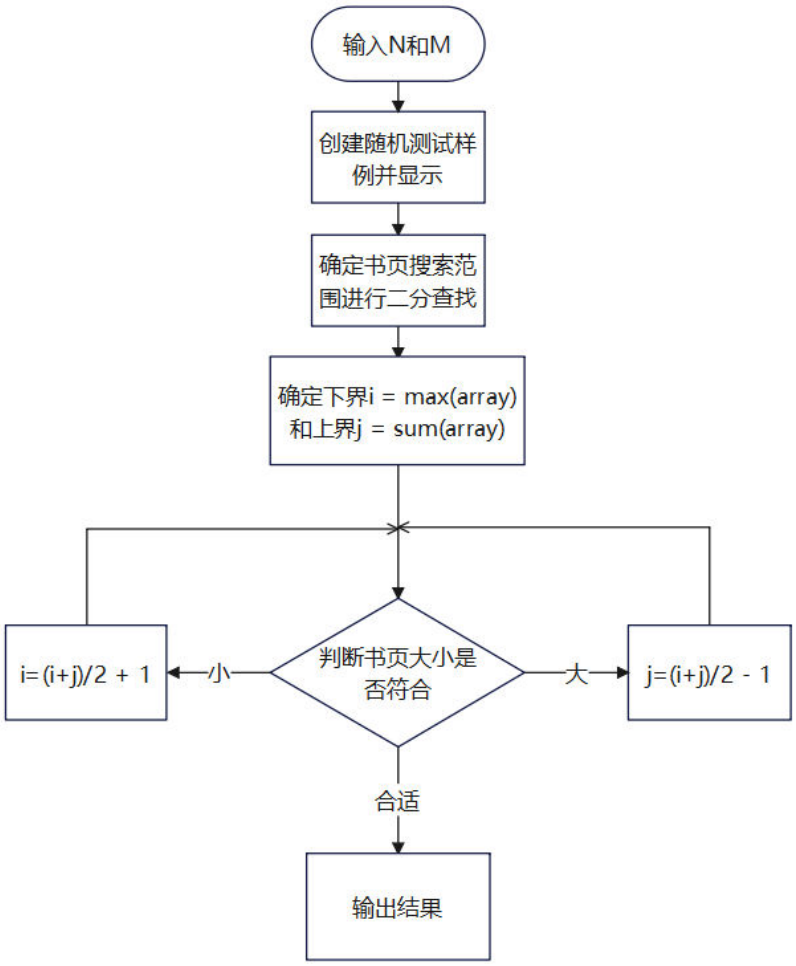
- 上述搜索策略提到了对容量进行判定，如何进行判定不能简单地比较大小，需要利用给定的容量遍历数组进行累加判定，会有以下几种情况：

    1. 如果当前累加值大于给定容量，则将count++，累加值归零
    2. 如果当前累加值小于给定容量
        1. 如果遍历指针没有指向数组末尾，则累加值加上当前数组值
        2. 如果遍历指针指向数组末尾，则将count++(特判)
    3. 如果当前累加值等于给定容量，则表示有可能找到解，设置flag暂存

- 经过上面的过程，最后对规定的学生个数和count值进行比较：

    1. 如果count < M 则表示当前容量偏大
    2. 如果count > M 则表示当前容量偏小
    3. 如果count == M 有两种情况：
        1. 如果flag被更改过，则表示找到解
        2. 如果flag是初始值，则表示偏大

- 流程图：


输入N和M
创建随机测试样例并显示
确定书页搜索范围进行二分查找
确定下界i = max(array)和上界j = sum(array)
判断书页大小是否符合
小
i=(i+j)/2 + 1
大
j=(i+j)/2 - 1
合适
输出结果

源代码及注释

```
#include <iostream>
#include <algorithm>
#include<cstdlib>
#include<ctime>
#include<vector>

using namespace std;

// 创建解决问题的类
class Solution
{
public:

    // 生成随机测试样例
    vector<int> generate_sample(int N, int range)
    {
        srand((int)time(NULL));
        vector<int> store;
        for(int i=0;i<N;i++)
        {
            store.push_back(rand() % range);
        }
        sort(store.begin(),store.end());
        return store;
    }

    // 利用二分查找搜索最小的书本页数容量
    int func(vector<int>& store, int M)
    {
        int sum = 0;
        for(auto& c:store)sum += c;
        int start = store[store.size() - 1];
        int end = sum;
        int i = start, j = end;
        while(i < j)
        {
            int volumn = (i + j) / 2;
            int flag = match(store, M, volumn);
            if(flag == 1)j = volumn - 1;
            else if(flag == -1) i = volumn + 1;
```

```
            else if(flag == 0) return volumn;
        }
        return i;
    }

    // 判断在给定的书页容量下每位同学是否都能容纳
    int match(vector<int>& store, int M, int volumn)
    {
        int count = 0;
        int flag = 1;// flag值表示当前容量的大小，如果为 -1 则偏小，如果为 0 则正好，如果为 1 则偏大
        int temp = 0;
        // 遍历数组
        for(int i=0;i<store.size();i++)
        {
            // 当前累加值小于容量
            if(temp < volumn - store[i])
            {
                if(i < store.size() - 1)temp += store[i];
                else if(i == store.size() - 1)count++; // 对末位进行特判
            }
            // 当前累加值大于容量
            else if(temp > volumn - store[i])
            {
                count++;
                temp = 0;    // 重置累加值
                i--;         // 指针回退
            }
            // 当前累加值等于容量
            else if(temp == volumn - store[i])
            {
                count++;
                temp = 0;    // 重置累加值
                flag = 0;    // 表示当前正好符合，找到了答案
            }
        }
        if(count == M)
        {
            if(flag == 0)return 0;
            else return 1;
        }
        else if(count < M) return 1;
```

```
        else return -1;
    }


    // 数组打印函数
    void print_array(vector<int>& store)
    {
        cout<<"The random sample array is: ";
        for(auto& c:store)cout<<c<<' ';
        cout<<endl;
    }

};

int main()
{
    class Solution s;
    cout<<"Please input the N and M:\n";
    int N, M;
    cin>>N>>M;  //输入N和M值
    vector<int> store = s.generate_sample(N, 100);
    //store = {12, 34, 67, 90}; //标准测试样例
    s.print_array(store);
    int res = s.func(store, M);
    cout<<"The result is "<<res<<endl;
    return 0;
}
```

实验结果展示及分析

实验结果展示

- 标准测试

```
PS D:\桌面\算法分析与设计> cd "d:\桌面\算法分析与设计\" ;
Please input the N and M:
4 2
The random sample array is: 12 34 67 90
The result is 113
PS D:\桌面\算法分析与设计>
```

- 三次随机测试

```
PS D:\桌面\算法分析与设计> cd "d:\桌面\算法分析与设计\"
Please input the N and M:
4 2
The random sample array is: 5 17 25 86
The result is 86
PS D:\桌面\算法分析与设计>
```

```
PS D:\桌面\算法分析与设计> cd "d:\桌面\算法分析与设计\"
Please input the N and M:
5 3
The random sample array is: 12 38 38 69 86
The result is 88
PS D:\桌面\算法分析与设计>
```

```
PS D:\桌面\算法分析与设计> cd "d:\桌面\算法分析与设计\"
Please input the N and M:
6 4
The random sample array is: 7 27 54 58 67 84
The result is 88
PS D:\桌面\算法分析与设计>
```

实验结果分析

- 正确性分析： 通过对实验结果的观察，发现标准测试和三次随机测试都表现出正确的结果，即证明算法的设计和实现的正确性
- 复杂度分析： 对于每次测试样例，采用二分搜索的方式，搜索范围为[max(array),sum(array)]，计为 m ， O(m) <= O(N * max(array))并且每次判别都再次遍历了原来的数组，因此**总的时间复杂度为O(nlog(m))**