算法分析与设计

第一章

1020383827@qq.com

doubleh

解题报告评分标准

- 在算法设计习题课件第一章的习题中，至少挑选出1题在soj上通过并写成解题报告，解题报告中说明通过题目的题号（使用屏幕截图列出）与数量，并包含至少1题的详细解题思路。在4月1日前把解题报告提交至ftp://172.18.57.223/，帐号为smie，密码为student001.
- 每一次解题报告满分为10分
  - 所写解题报告题数 = 规定的最少题目数，最高得8分
  - 多写一题，多1分，上限为10分
  - 抄袭0分，迟交80%分
- 交到对应章节文件夹，命名方式chapter1_12345678_zhangsan.zip

  (要求包含所有cpp文件和pdf解题报告)

- 1035 DNA matching
- 1198 Substring
- 1093 Air Express
- 1438 Shopaholic
- 1681 Matchsticks
- 10359 Valuable Jewellery
- 1405 Mahershalalhashbaz, Nebuchadnezzar, and Billy Bob Benjamin Go to the Regionals
- 1620 SCVs and minerals
- 1783 Large is Better
- 2503 最长字符串
- 8536 Happy Camper
- 6771 Class Packing

- **1035 DNA matching**
- **1198 Substring**
- 1093 Air Express
- 1438 Shopaholic
- **1681 Matchsticks**
- **10359 Valuable Jewellery**
- 1405 Mahershalalhashbaz, Nebuchadnezzar, and Billy Bob Benjamin Go to the Regionals
- **1620 SCVs and minerals**
- **1783 Large is Better**
- **2503** 最长字符串
- 8536 Happy Camper
- 6771 Class Packing

soj 1035 DNA matching

- 题意：给n个DNA单链，问最多能组成多少对DNA双链，每一个DNA单链只能使用一次，并且不能够翻转。两个DNA单链能够形成一对DNA双链的条件是对应位置A/T，C/G配对（碱基互补配对原则）
- 约束：1 <= n <= 100，DNA单链的长度m <= 100多组数据

soj 1035 DNA matching

- 贪心就好了！

- 举个例子：
  - 给定以下5个DNA单链：ATCG、TAGC、TAGG、TAGC、ATCC
  - 可以组成2个DNA双链：

    ATCG和TAGC匹配成双链1、TAGG和ATCC匹配成双链2

- 分析：对于每一个DNA单链，只能匹配一种特定的DNA双链，那么我们只需要贪心地做匹配就行

soj 1035 DNA matching

- 解法1：两重循环，对于每一个DNA单链找一个没有被匹配的，并且能与当前DNA单链匹配的DNA单链做匹配
- O(n^2*m)

soj 1035 DNA matching

- 解法2：数据结构加速O(nmlogn)
- 代码：

```
48 int matching(int n)
49 {
50     static char match[127];
51     match['A'] = 'T';
52     match['C'] = 'G';
53     match['G'] = 'C';
54     match['T'] = 'A';
55
56     int ret = 0;
57     multiset<string> S;
58     string str, match_str;
59     for (int i = 0; i < n; i++)
60     {
61         cin >> str;
62         match_str = "";
63         for (int j = 0; j < str.size(); j++)
64             match_str += match[str[j]];
65         if (S.count(match_str))
66         {
67             ret++;
68             S.erase(S.lower_bound(match_str));
69         }
70         else
71             S.insert(str);
72     }
73     return ret;
74 }
```

soj 1198 Substring

- 题意：给n个字符串，现在需要将他们拼接起来形成一个字典序最小的字符串
- 约束：1 <=n <= 8

- 举个例子：
  - 给定3个字符串：a、ab、ac
  - 拼接起来形成字典序最小的字符串是：aabac

- n个字符串按字典序从小到大排序，再拼起来？

soj 1198 Substring

- 一个很“显然”但是错误的解法：对n个字符串按字典序进行排序，之后从小到大得拼接起来

- 反例：b，ba，按字典序从小到大，则b排在ba前面，拼接会形成bba，而事实上bab才是拼接后字典序最小的字符串。

soj 1198 Substring

- 解法1：枚举所有不同拼接方式O(n!)
- 两种实现：
- 1）dfs枚举
- 2）使用stl函数next_permutation，注意前面的排序

```
113 string substring(int n, string str[])
114 {
115     sort(str, str + n);
116     string ret = "";
117     do
118     {
119         string tmp;
120         for (int i = 0; i < n; i++)
121             tmp += str[i];
122         if (ret == "" || tmp < ret)
123             ret = tmp;
124     } while (next_permutation(str, str + n));
125     return ret;
126 }
```

http://www.cplusplus.com/reference/algorithm/next_permutation/?kw=next_permutation

Rearranges the elements in the range [first,last) into the next lexicographically greater permutation.

soj 1198 Substring

- 解法2：贪心？
- 修正比较函数

```
128 bool cmp(const string &lhs, const string &rhs)
129 {
130     return lhs + rhs < rhs + lhs;
131 }
```

soj 1198 Substring

- 贪心证明：
- 这类贪心算法（含重载比较符）需要证明其传递性即
- 若a <cmp b，b <cmp c，那么a <cmp c

$$a \leq_{cmp} b \;\&\&\; b \leq_{cmp} c$$

- 证：

$$\iff ab \leq ba \;\&\&\; bc \leq cb$$

$$\boxed{\implies ac \leq ca}$$

思考：这一步怎么推出来，写个证明。

$$\iff a \leq_{cmp} c$$

Soj 1093 Air Express

- 题意：给出4个重量区间和各个区间的单位重量运输价，问对于一个背包，需要添加多少重量使得运输代价最小。
- 约束：所有的数都为正数，且不超过1000

Soj 1093 Air Express

- 解法：考虑到取得最低代价的情况：1）不加重量 2）加重量到一个重量区间的下界，直接枚举就好了

```
int air_epress(int lowwer[4], int upper[4], int price[4], int x)
{
    int ret;
    for (int i = 0; i < 4; i++)
    {
        if (x >= lowwer[i] && x <= upper[i])
            ret = x * price[i];
        else if (x < lowwer[i])
            ret = min(ret, lowwer[i] * price[i]);
    }
    return ret;
}
```

Soj 1438 Shopaholic

- 题意：有个购物狂在商场中购物，他想要买n个商品，商场在做促销，每买三个商品，最便宜的一个可以免费，现在问最多可以省多少钱。
- 约束：n<=2*10^4 price<=2*10^4

Soj 1438 Shopaholic

- 解法：贪心
- 我们可以很明显的发现我们我们将最贵的三个作为一组，次贵的三个作为一组，…
- 这样我们能省下第3贵、第6贵、…的，这个时候能够省下最多的钱
- greater<int>()是一个模板类的
- 构造函数，被称为谓语

```
int shopaholic(int price[], int n)
{
    sort(price, price + n, greater<int>());
    int ret = 0;
    for (int i = 2; i < n; i += 3)
        ret += price[i];
    return ret;
}
```

Soj 1438 Shopaholic

- greater<int>()是一个模板类的构造函数，被称为谓语，是一种用于自动生成简单比较函数的方法。
- 其中greater意思是从大到小排序，这个需要所指类型具有或者重载了大于号。
- 一个要成为谓语需要重载其括号操作符（）

```
int shopaholic(int price[], int n)
{
    sort(price, price + n, greater<int>());
    int ret = 0;
    for (int i = 2; i < n; i += 3)
        ret += price[i];
    return ret;
}
```

soj 1681 Matchsticks

- 题意：拥有n根火柴，求出可以摆出的最大和最小的数

- 分析：给定一个数，可以确定所需的火柴数，所以可以枚举少部分的情况找规律

数字	1	2	3	4	5	6	7	8	9	0
所需火柴数	2	5	5	…						

soj 1681 Matchsticks

总的火柴数	Min （数字8消耗的火柴数最多）	Max （数字1消耗火柴数最少）
…	…	…
8	10	1111
9	18	7111
10	22	11111
11	20	71111
12	28	111111
13	68	711111
14	88	1111111
15	108	7111111
16	188	11111111
…	…	…

soj 1681 Matchsticks

- 解法：由于1所消耗的火柴是最少的，所以最大数一定是大量的使用火柴摆成”1”，如果落单了的话，那么就把开头的那个”1”变成”7”

- 最少数的时候，在火柴充分的情况下，由于”8”所消耗的火柴是最多的，因此会用大量的火柴摆成”8”填到末尾，之后少数火柴的情况下直接枚举答案 (by poetry)

- 要考虑一些特殊的情况

soj 10359 Valuable Jewellery

- 题意：有N个物品和K个背包，物品有质量和价值两个属性，每个背包有对应的最大承受质量，一个背包只能装一个物品，询问最大能够带走的价值。
- 解法：背包从小到大排序，物品按照质量从小到大排序。先考虑小的背包，对于当前考虑的背包，选择能够装入背包且最贵的物品装入。
- 使用到stl中的priority_queue，sort进行优化
- http://www.cplusplus.com/reference/queue/priority_queue/priority_queue/

soj 1405 Mahershalalhashbaz, Nebuchadnezzar, and Billy Bob Benjamin Go to the Regionals

- 题意：给出参数n和k，n为人数，k为所分组的每组的人数(这样便得到n/k个组)。然后需要满足以下条件：组内所有人的人名长度，和这个组的平均人名长度的差距不大于2(平均长度在这里可以是有小数的)，问是否存在这样的分组方法。
- 解法：物以类聚，人以群分，我们获取每个人的人名长度后，将其排序，然后k个数过来分成一组，这样就尽可能的保证差距。之后计算即可。
- 可用乘法代替直接比较，避免精度误差
- 可以用前缀和进行效率上的优化 (by poetry)

soj 1620 SCVs and minerals

- 题意：一开始你有N个农民和M单位的资源，每个农民一秒钟可以得到C单位的资源，每个农民可以用P单位的资源立即生产，周围资源没有限制，没有人口限制（开了作弊？）,而且农民采集资源的过程是离散的(例如不会在0.5秒的时候收入0.5C),问在S秒后所收集的最大资源是多少（不包括已经花出去的）。
- 解法：
  - 第i秒产生的农民可以在剩下的时间内产生 $(S-i+1)\times C$ 单位资源
  - 这个农民的成本是P单位资源
  - 如果 $(S-i+1)\times C > P$，即选择在这一秒用资源产生尽可能多的农民，否则不消耗资源去产生农民
- 前期尽可能消耗资源产生农民，后期等待即可。

soj 1783 Large is Better

- 题意：给出一个数，你可以对这个数做出以下调整：交换任意两个相邻的数，没有次数限制，交换的两个数中不能有0，求可以得到的最大的数。

- 举个例子：
  - 给定数1012400198
  - 可以得到的最大的数是1042100981

- 解法：按照0作为分割处，将每段数都按照从大到小排序，最后输出。(by poetry)

soj 2503 最长字符串

- 题意：
  - 要求你构造一个由字符'A', 'B'组成的字符串, 满足以下几个条件:
    - 1) A的个数<=countA
    - 2) B的个数<=countB
    - 3) 连续的A的个数不可以超过maxA.
    - 4) 连续的B的个数不可以超过maxB.
    - 5) 这个字符串的长度最长.
    - 给你countA,countB,maxA,maxB,要求你输出字符串的最大长度.
- 题解：分别以A、B为间隔符进行考虑，取两种情况中的最大值
  - 以A为间隔符，考虑countA与B所形成的间隔位置数[]*BBB*[]*BBB*[]*B*[]
  - 以B为间隔符，考虑countB与A所形成的间隔位置数[]*AAA*[]*AAA*[]*A*[]

soj 8536 Happy Camper

- 题意：在连续的P天里可以有L天享受假期，这P天不能和另一个阶段的P天重用，问在V天里最多享受的假期数。
- 解法：直接将V天按P天分成若干的间隔，然后每个间隔尽量享受最多的假期 (by poetry)

soj 6771 Class Packing

- 题意：有年级为0,1,2,3,4,5,6共七个年级的孩子，每个年级的孩子数量给出。现为这些孩子分配班级，满足所分配班级的个数最小。其中，分配班级的时候需要满足以下条件：

  1：一个班级只能有一个年级，或者两个相邻年级的孩子

  2: 拥有年级0-2的孩子的班级，其人数最多为20

  3: 拥有年级3-4的孩子的班级，其人数最多为25

  4: 拥有年级5-6的孩子的班级，其人数最多为30

soj 6771 Class Packing

- 题解：考虑年级为0的孩子，显然如果人数多的话，将他们尽量分配到同一个班的结果是理想的（因为如果分配到其他的班，可能会影响到高年级的班级的容量），如果人数有剩下的话，就从其上一年级中调派人员过来，使得该班的容量填满，这样便是年级为0的孩子分配的一个最优的情况。
- 对于其他年级，也是这么处理。(by poetry)