****



**程序设计综合实践课程报告**

**基础算法1实验**

**学生姓名**

**学院名称 智能与计算学部**

**专 业 工科试验班(智能与计算类)**

**学 号**

# 1. 闰年判断

## 1.1题目分析

**能被4整除且不能被100整除，或者能被400整除的年份是闰年。**

## 1.2 题目代码（带注释）

|  |
| --- |
| #include <iostream>  using namespace std;    int main()  {      int n, y;      cin >> n;      while (n--) {          cin >> y;          if ((y % 4 == 0 && y % 100 != 0) || y % 400 == 0)              cout << "Yes" << endl;          else              cout << "No" << endl;      }      return 0;  } |
|  |

# 2. 这是第几天

## 2.1题目分析

**当前月之前所有月的天数，加上当前月内日期之前的天数，注意判断闰年。**

## 2.2 题目代码（带注释）

|  |
| --- |
| #include <iostream>  using namespace std;    bool isLeap(int y)  {      if ((y % 4 == 0 && y % 100 != 0) || y % 400 == 0)          return true;      else          return false;  }  int main()  {      int n;      cin >> n;      while (n--) {          int y, m, d;          cin >> y >> m >> d;          int count = 0;          int month[12] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};          if (isLeap(y))              month[1] = 29;          for (int i = 1; i < m; i++)              count += month[i - 1];          count += d;          cout << count << endl;      }      return 0;  } |
|  |

# 3. 机器人模拟

## 3.1题目分析

**用switch判断依次处理机器人移动的方向。**

## 3.2 题目代码（带注释）

|  |
| --- |
| #include <iostream>  using namespace std;    int main()  {      int n;      cin >> n;      while (n--) {          string track;          int x = 0, y = 0;          cin >> track;          for (int i = 0; i < track.length(); i++) {              switch(track[i]) {                  case 'W': {y++;break;}                  case 'S': {y--;break;}                  case 'A': {x--;break;}                  case 'D': {x++;break;}              }          }          cout << x << " " << y << endl;      }      return 0;  } |
|  |

# 4. 页码统计

## 4.1题目分析

**无论页码是多少都是1～n，所以可以从1到n进行遍历并对每个数进行分解。**

## 4.2 题目代码（带注释）

|  |
| --- |
| #include <iostream>  #include <vector>  using namespace std;    void cnt(int k, int\* count)  {      vector<int>dig;      while (k) {          dig.push\_back(k % 10);          k /= 10;      }      for (int i = 0; i < dig.size(); i++)          count[dig.at(i)]++;  }  int main()  {      int n;      cin >> n;      while (n--) {          int k;          int count[10] = {0};          cin >> k;          for (int i = 1; i <= k; i++) {              cnt(i, count);          }          cout << count[0];          for (int i = 1; i < 10; i++)              cout << " " << count[i];          cout << endl;      }      return 0;  } |
|  |

# 5. 顺子

## 5.1题目分析

**见代码。**

## 5.2 题目代码（带注释）

|  |
| --- |
| #include <iostream>  using namespace std;    int main()  {      int T;      cin >> T;      while (T--) {          int n, a, b, i, sum = 0;          int poker[14];          for (i = 1; i <= 13; i++)              poker[i] = 4;          cin >> n;          for (i = 0; i < n; i++) {              cin >> a >> b;              if (poker[b] > 0)                  poker[b]--;          }          for (i = 1; i <= 11; i++) {              int \_sum = 1;              if (poker[i] && poker[i+1] && poker[i+2]) {                  \_sum \*= (poker[i] \* poker[i+1] \* poker[i+2]);                  sum += \_sum;              }          }          cout << sum << endl;      }      return 0;  } |
|  |

# 6. 字符串

## 6.1题目分析

**将短字符串的第一个字符依次与长字符串的字符比较，并记录位置，如果一样则让长短字符串字节往后移一位，继续比较，用flag作状态参数，如果最后是1则输出所在位置，为0代表不存在子串，输出-1。**

## 6.2 题目代码（带注释）

|  |
| --- |
| #include <iostream>  using namespace std;    int main()  {      int T;      cin >> T;      while (T--) {          string supstr, str;          cin >> supstr >> str;            int next[str.length()];          next[0] = -1;          int i = 0, j = -1;            while (i < str.length()) {              if (j == -1 || str[i] == str[j]) {                  i++;                  j++;                  next[i] = j;              } else                  j = next[j];          }            i = 0;          j = 0;          while (i < supstr.length() && (j == -1 || j < str.length())) {              if (j == -1 || supstr[i] == str[j]) {                  i++;                  j++;              } else                  j = next[j];          }          if (j == str.length())              cout << (i - j) << endl;          else              cout << -1 << endl;      }      return 0;  } |
|  |

# 7. 区间和统计

## 7.1题目分析

**利用循环，实现数字的累加。留下总和等于所要数字的组即可**

## 7.2 题目代码（带注释）

|  |
| --- |
| #include <iostream>  using namespace std;  int main()  {      int T;      cin >> T;        while (T--) {          int n, p;          cin >> n >> p;          int a[n];          for (int i = 0; i < n; i++)              cin >> a[i];          int count = 0;            for (int i = 0; i < n; i++)          {              int sum = 0;              for (int k = 0; k < n - i; k++)              {                  sum += a[i + k];                  if (sum == p)                      count++;              }          }          cout << count << endl;      }  } |
|  |

# 8. 斐波那契数列

## 8.1题目分析

**利用递归超时，选择用循环的方式。**

## 8.2 题目代码（带注释）

|  |
| --- |
| #include <iostream>  using namespace std;    const int N = 61;    int main()  {      int T;      cin >> T;      while (T--) {          int n, i;          long int fib[N];          fib[0] = 0;          fib[1] = 1;          cin >> n;          if (n > 1)              for (i = 2; i <= n; i++)                  fib[i] = fib[i-1] + fib[i-2];          cout << fib[n] << endl;      }      return 0;  } |
|  |

# 9. 汉诺塔

## 9.1题目分析

**利用递归超时，选择用循环的方式**

## 9.2 题目代码（带注释）

|  |
| --- |
| #include <iostream>  using namespace std;    const int N = 41;    int main()  {      int T;      cin >> T;      while (T--) {          int n, i;          long int hanoi[N]; // LOOOOOOONG!!!          hanoi[1] = 1;          cin >> n;          if (n > 1)              for (i = 2; i <= n; i++)                  hanoi[i] = 2 \* hanoi[i-1] + 1;          cout << hanoi[n] << endl;      }      return 0;  } |
|  |