南京大学
NANJING UNIVERSITY

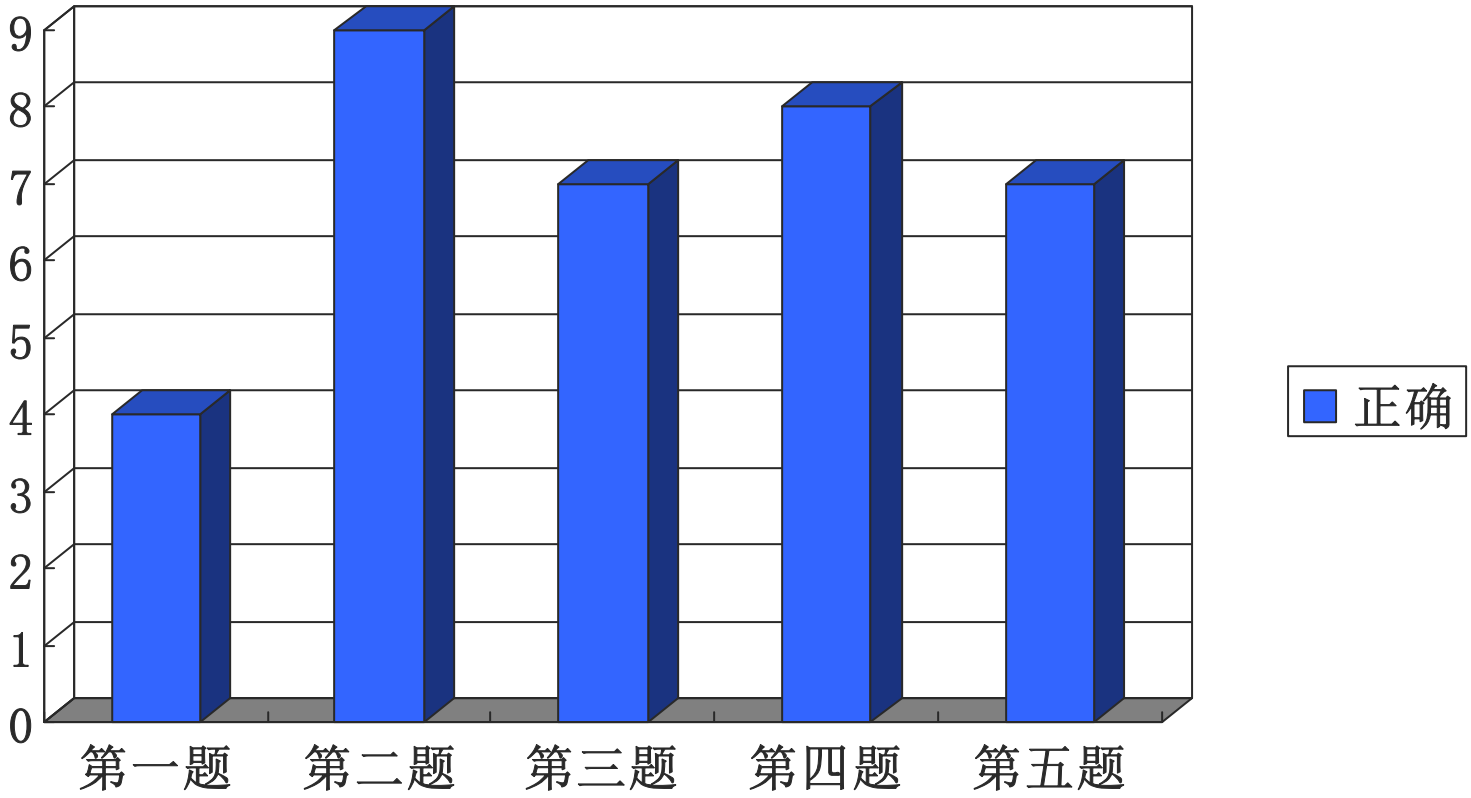
测试反馈和编程规范

程序测试结果

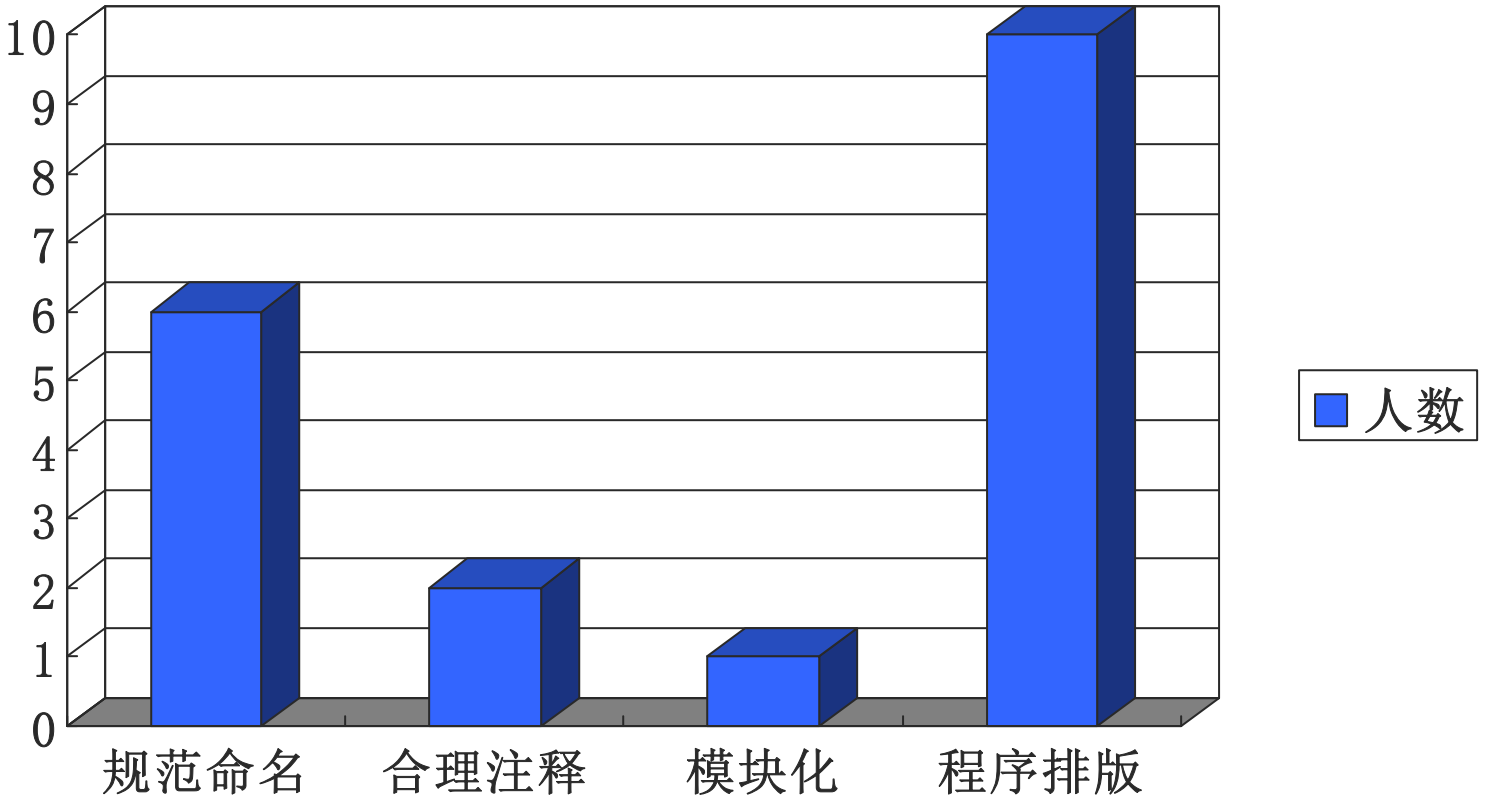
7位同学：一张白纸能够画出最美的图案！☺

- 13位同学提交程序
- 2位同学5道题目全部做对
- 5位同学做对4道
- 1位同学使用模块化的思想解决第5道题
- 2位同学程序中使用了注释
- 6位同学有变量规范命名的意识

程序测试结果



程序测试结果



第一道题 例子1

1. 简洁

```
#include <cstdio>
int a,b,n,i,x;
int main() {
    scanf("%d%d",&n,&x);
    a=b=x;
    for (i=1;i<n;i++) {
        scanf("%d",&x);
        if (a<x) a=x;
        if (b>x) b=x;
    }
    printf("%d %d\n",a,b);
    return 0;
}
```

第一道题 例子2

```
#include <iostream>
#include <cstdio>
#include <cstring>
#include <algorithm>
using namespace std;
int n,in,nmax,nmin;
int main(){
        scanf("%d",&n);
        scanf("%d",&in);
        nmax=nmin=in;
        for(int i=1; i<n; i++){
                scanf("%d",&in);
                nmax=max(nmax,in);
                nmin=min(nmin,in);
        }
        printf("%d %d",nmax,nmin);
        return 0;
}
```

变量名命名有含义

第一道题 例子3

```
# include <iostream>
using namespace std;
const int MAXN=1000001;
int n,max_num,min_num;
int num[MAXN];
void init()//读入n个数
{
    int i;
    cin>>n;
    for (i=1;i<=n;i++)cin>>num[i];
}
void work()
{
    int i;
    max_num=0;min_num=MAXN;
    for (i=1;i<=n;i++)
    {
        if (max_num<num[i])max_num=num[i];//找最大值
        if (min_num>num[i])min_num=num[i];//找最小值
    }
    cout<<max_num<<' '<<min_num<<endl;//输出结果
}
int main ()
{
   init();
   work();
   cout<<endl;
   return 0;
}
```

有模块化的意识

第二道题 例子1

```
# include <iostream>
using namespace std;
int n;
void work()
{
    int i,j;
    cin>>n;
    for (i=1;i<=n;i++)//输出前n行
    {
       for (j=1;j<=n-i;j++)cout<<' ';//补空格
       for (j=1;j<=2*i-1;j++)cout<<'*';
       for (j=1;j<=n-i;j++)cout<<' ';//补空格
       cout<<endl;
    }
    for (i=n-1;i>=1;i--)//输出后n-1行
    {
       for (j=1;j<=n-i;j++)cout<<' ';//补空格
       for (j=1;j<=2*i-1;j++)cout<<'*';
       for (j=1;j<=n-i;j++)cout<<' ';//补空格
       cout<<endl;
    }
}
int main ()
{
   work();
   cout<<endl;
   return 0;
}
```

有注解！

模块化，结构清晰！

第二道题 例子2

```
#include <cstdio>

int abs(int x)
{return x>0?x:-x;}

int main()
{
    int n;
    scanf("%d",&n);
    for (int i=1;i<n<<1;i++)
    {
        for (int j=1;j<=abs(n-i);j++) printf(" ");
        for (int j=1;j<=(n<<1)-(abs(n-i)<<1)-1;j++) printf("*");
        printf("\n");
    }
    return 0;
}
```

简洁的做法！

第三道题 例子1

```
#include <iostream>
using namespace std;
int huiwen(long n){
        int i,a[10];
        a[0]=0;
        while (n!=0) {
                a[0]++;
                a[a[0]]=n%10;
                n=n/10;
        }
        for (i=1;i<=a[0]/2;i++) {
                if (a[i]!=a[a[0]-i+1]) return 1;
        }
        return 0;
}
int main(){
        long n,i,s;
        cin>>n;
        s=0;
        i=1;
        while (s<n) {
                if (huiwen(i*i)==0) {
                        cout<<i*i<<endl;
                        s++;
                }
                i++;
        }
        return 0;
}
```

模块化思想；使用long类型

第三道题 例子2

```
#include<iostream>
#include<cstring>
#include<cstdio>
#include<cstdlib>
#include<algorithm>
using namespace std;
int i,j,k,m,n,tot;
int a[11];
bool flag;

int main() {
  cin>>n;
  tot=0;

  for (i=1;tot<n;i++) {
    k=i*i;
    j=0;
    while (k) {
      a[++j]=k%10;
      k/=10;
    }
    flag=false;
    for (k=1;k<=j>>1;k++)
      if (a[k]!=a[j+1-k]) {flag=true;break;}
    if (!flag) {
      cout<<i*i<<endl;
      tot++;
    }
  }
  return 0;
}
```

熟练使用移位操作；简洁的程序

第三道题 例子3

```
# include <iostream>
# include <string>
using namespace std;
int n;
int isok(int num)//判断num是否回文
{
   char c;
   string s1,s2;
   s1=s2="";
   while (num!=0)
   {
        c=(num%10)+'0';
        s1=s1+c;s2=c+s2;//s1为num的原序，s2为s1的逆序
        num/=10;
   }
   if (s1==s2)return 1;//因为回文数正着读反着读一样
   else return 0;
}
void work()
{
    int i,num;
    i=1;num=1;
    cin>>n;
    while (i<=n)//这里我们可以枚举平方和减少计算量优化
    {
       if (isok(num*num))
       {
          cout<<num*num<<endl;//输出结果
          i++;
       }
       num++;
    }
}
int main ()
{
   work();
   cout<<endl;
   return 0;
}
```

有注解

第四道题 例子1

```
# include <iostream>
# include <string>
using namespace std;
int m,n;
string ans="";
void work()//把m转换成n进制
{
    int num;
    char c;
    cin>>m>>n;
    while (m>0)
    {
        num=m%n;
        if (num<=9)c=num+'0';//判断是否需要转换成字母表示
        else c=num-10+'A';
        ans=c+ans;//ans为结果值
        m/=n;
    }
    cout<<ans;
}
int main ()
{
   work();
   cout<<endl;
   return 0;
}
```

模块化，结构清晰，有注解！

第五道题 例子

```
#include <iostream>
using namespace std;
int huiwen(long n,long k){
        int i,a[10];
        a[0]=0;
        while (n!=0) {
                a[0]++;
                a[a[0]]=n%k;
                n=n/k;
        }
        for (i=1;i<=a[0]/2;i++) {
                if (a[i]!=a[a[0]-i+1]) return 1;
        }
        return 0;
}
int main(){
        int n,s;
        cin>>s;
        while ((huiwen(s,10)==1)|(huiwen(s,2)==1))
                s++;
        cout<<s;
        return 0;
}
```

程序简洁；模块化；容易懂。

一个好的程序

- 排版、注释、可读性
- 变量、结构体、函数、过程
- 可测试性、程序效率、质量保证
- 代码编辑、编译、审查
- ……

排版、注释、可读性

1.程序块采用缩进风格编写，缩进的空格数为4个。
2.对齐只使用空格键，不使用TAB键。
2.避免多个短语句写在一行，即一行只写一条语句。
3.一行程序以小于80字符为宜，不要写得过长。
4.注释格式尽量统一，建议使用/* */
5.源文件头部和函数头部应进行注释。
6.不应直接使用数字，要用有意义的枚举或宏来代替。
7.注意运算符的优先级，用括号明确表达式的操作顺序

……

变量、结构体、函数、过程

1.严禁使用未经初始化的变量作为右值。

2.防止局部变量与全局变量同名 。

3.结构体的功能要单一，是针对一种事务的抽象。

4.函数名应准确描述函数的功能，使用动宾词组为执行某操作的函数命名 (e.g. GetCurrentTime……)。

5.函数的规模尽量限制在200行以内。

6.检查函数所有参数输入的有效性。

7.避免函数中不必要语句，防止程序中的垃圾代码。

……

可测试性、程序效率、质量保证

1.使用断言来发现软件问题，提高代码可测性 。

2.循环体内工作量最小化。

3.尽量减少循环嵌套层次。。

4.在保证程序的正确性、稳定性及可读性的前提下，提高代码效率。

5.时刻注意表达式是否会上溢、下溢(e.g. unsigned)。

6.有可能的话，if语句尽量加上else分支，对没有else分支的语句要小心对待；switch语句必须有default分支

……

推荐的书籍

1 《Clean Code》Robert C. Martin

2 《Google C++ Style Guide》Benjy Weinberger

3 《高质量C/C++编程指南》林锐著

Q&A

我们的程序

- int s,m,i,j,k,a[100];
- int huiwen(…){…}
- yushu=j%(10^cishu);
- Int k; … int main(){ for(i=k; i>0; i--) …}
- Bool check(..){…} bool check2(…){…}

我们的程序

- 有意义的常量
- const int maxLen=33;
- 模块化的思想
- while ((huiwen(s,10)==1)|(huiwen(s,2)==1))
- 使用注释
- int isok_B(int m)//判断十进制数m在二进制下是否回文