作业 1 参考答案（by 宋天恩）

P23-16

```
#include <iostream>
#include <sstream>
using namespace std;

// 正负
enum signType
{
   Plus,
   Minus
};

class Currency
{
public:
   Currency(signType theSign = Plus, unsigned long theDollars = 0, unsigned int theCents = 0)
   {
      setValue(theSign, theDollars, theCents);
   }
   ~Currency() {}
   // 原有函数，部分用不到的省略了
   void setValue(signType theSign, unsigned long theDollars, unsigned int theCents)
   {
      sign = theSign;
      dollars = theDollars;
      cents = theCents;
   }
   void setValue(double);
   signType getSign() const { return sign; }
   unsigned long getDollars() const { return dollars; }
   unsigned int getCents() const { return cents; }
   Currency add(const Currency &) const;
   Currency &increment(const Currency &);
   void output()
   {
      if (sign == Minus)
         cout << "-";
      cout << "$" << dollars << '.' << cents;
   }

   // 拓展函数(作业部分)
```

```
    void input();
    Currency subtract(const Currency &x) const;
    Currency percent(double x);
    Currency multiply(double x);
    Currency divide(double x);

private:
    signType sign;
    unsigned long dollars;
    unsigned int cents;
};

void Currency::input()
{
    cout << "输入格式：$1.09、-$3.91" << endl;
    string str;
    getline(cin, str);
    stringstream ss(str);
    if (str[0] == '-')
    {
        this->sign = Minus;
        char ch;
        ss >> ch >> ch >> dollars >> ch >> cents;
    }

    else
    {
        this->sign = Plus;
        char ch;
        ss >> ch >> dollars >> ch >> cents;
    }
}

Currency Currency::subtract(const Currency &x) const
{
    long a1, a2, a3;
    Currency result;
    a1 = dollars * 100 + cents;
    if (sign == Minus)
        a1 *= -1;

    a2 = x.dollars * 100 + x.cents;
    if (x.sign == Minus)
        a2 *= -1;
```

```
    a3 = a1 - a2;
    if (a3 < 0)
    {
        result.sign = Minus;
        a3 = -a3;
    }
    result.dollars = a3 / 100;
    result.cents = a3 % 100;

    return result;
}

Currency Currency::percent(double x)
{
    Currency result;
    long a = dollars * 100 + cents;
    a = a * x / 100;
    result.sign = sign;
    result.dollars = a / 100;
    result.cents = a % 100;
    return result;
}

Currency Currency::multiply(double x)
{
    Currency result;
    long a = dollars * 100 + cents;
    a = a * x;
    result.sign = sign;
    result.dollars = a / 100;
    result.cents = a % 100;
    return result;
}

Currency Currency::divide(double x)
{
    Currency result;
    long a = dollars * 100 + cents;
    a = a / x;
    result.sign = sign;
    result.dollars = a / 100;
    result.cents = a % 100;
    return result;
```

```
}

int main()
{
   // 测试用函数，可以写作其它函数测试
   Currency k(Plus, 3, 50);
   k.output();

   k.input();
   k.output();

   Currency tool(Plus, 1, 45);
   k = k.subtract(tool);
   k.output();

   k = k.percent(50);
   k.output();

   k = k.multiply(2.5);
   k.output();

   k = k.divide(2.5);
   k.output();
}
```

P29-23

```
#include <iostream>
using namespace std;
//(1)略
//(2)证明递归部分多次调用后可以转化成基础部分:
// 证:首先，y 不为 0，若 x%y 为 0，则下次递归就转化为了基础部分；若 x%y!=0,那么下次递归的 x 值就为 y,而 y 值肯定比本次的 y 值小（取模）
// 所以，可以保证在递归中，y 值是递减的，一定在有限次递归中会取到 0，既达到了基础部分。
//(3)代码实现:
int gcd(int x, int y)
{
   // 基础部分
   if (y == 0)
      return x;
   // 递归部分
   else
      return gcd(y, x % y);
```

```
}

int main()
{
    cout << gcd(20, 30) << endl;
    cout << gcd(112, 42) << endl;
}
```

P29-24

```
#include <iostream>
using namespace std;
int a[10000], n, x;
// 作业部分
bool query(int x, int pos)
{
    if (a[pos] == x)
        return true;
    if (pos == n - 1)
        return false;
    return query(x, pos + 1);
}

// 测试部分
int main()
{
    cout << "输入数组大小" << endl;
    cin >> n;
    for (int i = 0; i < n; i++)
    {
        a[i] = rand() % 10000;
        cout << a[i] << ' ';
    }
    cout << endl;

    cout << "输入想查询数据" << endl;
    cin >> x;
    cout << query(x, 0) << endl;
}
```