第四讲：基本的算法结构

姓名：丁保荣　　**学号**：171860509

2017 年 10 月 23 日

请独立完成作业，不得抄袭。
若参考了其它资料，请给出引用。
鼓励讨论，但需独立书写解题过程。

第一部分　作业

题目 (DH:2.1)

The algorithm for summing the salaries of N employees presented in the text performs a loop that consists of adding one salary to the total and advancing a pointer on the employee list N-1 times.The last salary is added separately. What is the reason for this? Why don't we perform the loop N times?

解答：

Because if we perform the loop N times, when the machine performs the N^{th} time, first we add the N^{th} employee's salary to the total, and then we point to the next of the N^{th} employee, namely the $(N+1)^{th}$ employee.

So here comes the problem.

The pointer excceeds the limitation of the number of the employees.

题目 (DH:2.2)

Consider the bubblesort algorithm presented in the text.

(a) Explain why the outer loop is performed only N-1 times.

(b) Improve the algorithm so that on every repeated execution of the outer loop,the inner loop checks one element less.

解答：

(a) After each outer loop,it places the largest number of the remaining unordered numbers at the beginning of the ordered numbers.

So when the last N-1 numbers are settled, the second number must be larger than the first number.

So there is no need to perform the N^{th} loop to settle the first number.

(b)

```
(1)set i=0;
(2)do the following N-1 times:
    (2.1) point to the first element;
    (2.2) do the following (N-1-i) times:
          (2.2.1) compare the element pointed to with the next element;
          (2.2.2) if the compared elements are in the wrong order, exchange them;
          (2.2.3) point to the next element;
    (2.3) let i++;
```

题目 (DH:2.3)

Prepare flowcharts for the bubblesort algorithm presented in the text and for the improved version you were asked to design in Exercise 2.2.

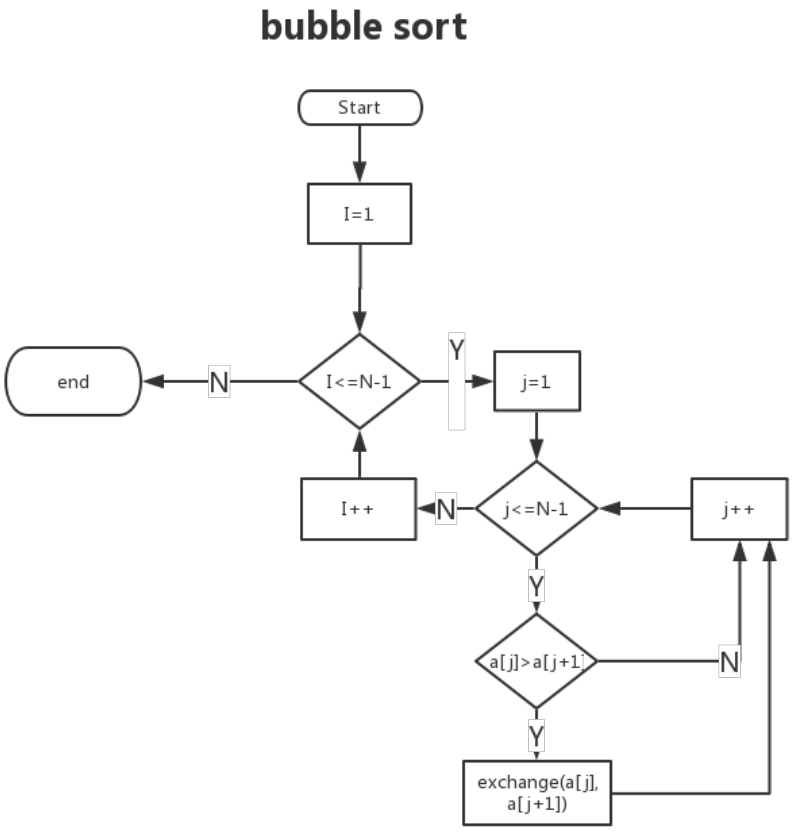

图 1: bubblesort

improved.png improved.bb

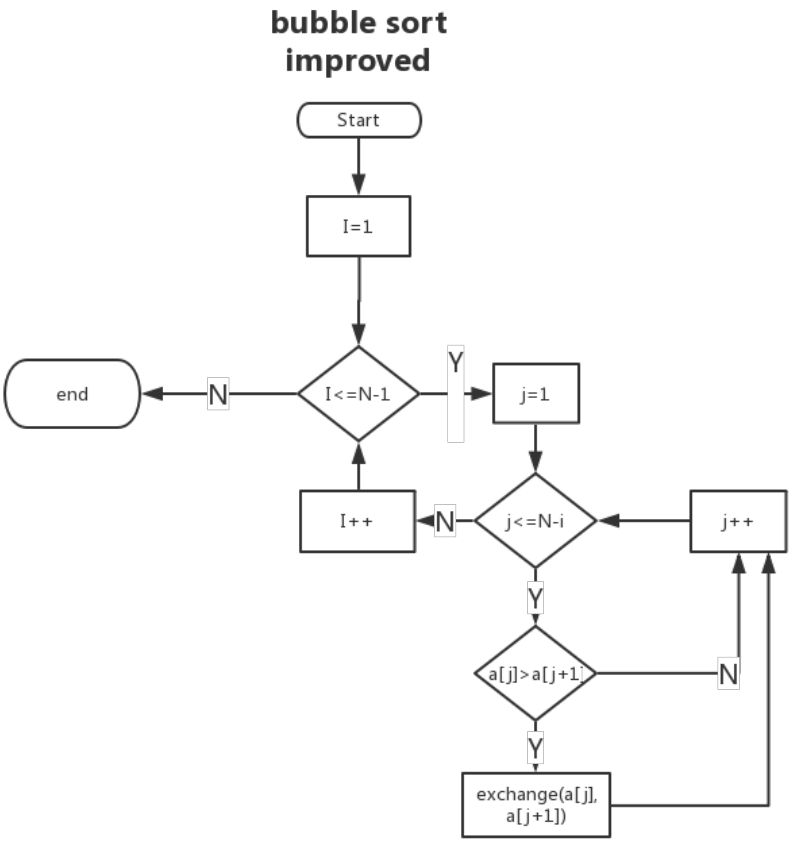

图 2: bubblesort improved

解答:

题目 (DH:2.4)

Write algorithms that ,given an integer N and a list L of N integers, produce in S and P the sum of the even numbers appearing in L and the product of the odd ones,respectively.
(a) Using bounded iteration.
(b) Using "goto" statements.

解答:

```
(a)
(1)set S=0;P=1;
(2) point to the first element;
(3) do the following N-1 times:
(3.1) if if the number is even then add the number to S,else mutiply the number to P;
(3.2) point to the next element;
(4) if the number is even then add the number to S,else mutiply the number to P;
(5) Print S and P;


(b)
(1)set S=0;P=1;
(2) point to the first element;
(3) do the following N-1 times:
(3.1) if if the number is even then goto (3.2),else goto(3.3);
(3.2) add the number to S; goto(3.4)
(3.3) mutiply the number to P;
(3.4) point to the next element;
(4) if the number is even then add the number to S,else mutiply the number to P;
(5) Print S and P;
```

题目 (DH: 2.5)

Show how to perform the following simulations of some control constructs by others. The sequencing construct "and-then"is implicitly available for all the simulations. You may introduce and use new variables and labels if necessary.
(a) Simulate a "for-do"loop by a "while-do"loop.
(b) Simulate the "if-then"and "if-then-else"statements by "while-do"loops.
(c) Simulate a "while-do"loop by "if-then"and "goto"statements.
(d) Simulate a "while-do"loop by a "repeat-until"loop and "if-then"statements.

解答:

(a) for (i=0;i<=N;i++) do A;

```
    i=0;
    while (i<=N)
    {
      do A;
      i++;
    }
```

(b) if (A) then

```
        do B;
    else
        do C;
```

```
    while(A)
    {
        do B;
    }
    while{not A}
    {
        do C;
    }
```

(c) while(A)

```
    {
        do B;
    }
```

```
    (1)if (A) then goto(2) else goto(3);
    (2) do B; goto (1);
    (3) done! exit;
```

(d) while(A)

```
    {
      do B;
    }
```

```
    if A then
    {
      repeat
      {
        do B
      }until(not A)
    }
\end{solution}
```

题目 (DH: 2.6)

Write down the sequence of moves resolving the Towers of Hanoi problem for five rings.

解答:

```
1  //
2  // main.cpp
3  // Towers of Hanoi
4  //
5  // Created by 丁保荣 on 2017/10/21.
6  // Copyright © 2017年 丁保荣. All rights reserved.
7  //
8
9  #include <iostream>
10 using namespace std;
11 void move(int start,int num, char from,char to,char via);
12 int main(void)
13 {
14     int n;
15     cin >> n;
16     move(1,n,'A','B','C');
17     return 0;
18 }
19
20 void move(int start, int num, char from, char  to, char via)
21 {
22     if(num==1)
23     {
24         cout <<"move " << from<<" to " << to<<endl;
25         return;
26     }
27     else
28     {
```

```
29          move(start,num-1,from,via,to);
30          cout<<"move "<< from<<" to "<<to <<endl;
31          move(start,num-1,via,to,from);
32      }
33  }
```

```
move A to B
move A to C
move B to C
move A to B
move C to A
move C to B
move A to B
move A to C
move B to C
move B to A
move C to A
move B to C
move A to B
move A to C
move B to C
move A to B
move C to A
move C to B
move A to B
move C to A
move B to C
move B to A
move C to A
move C to B
move A to B
move A to C
move B to C
move A to B
move C to A
move C to B
move A to B
```

题目 (DH: 2.7)

The factorial of a non-negative integer N is the product of all positive integers smaller than or equal to N. More formally, the expression N factorial, denoted by N!, is re-

cursively defined by $0! = 1$ and $(N+1)! = N! \times (N+1)$. For example, $1! = 1$ and $4! = 3! \times 4 = ... = 1 \times 2 \times 3 \times 4 = 24$.

Write algorithms that compute $N!$, given a non-negative integer N.

(a) Using iteration statements.
(b) Using recursion.

解答:

```
(a)
(1)set fac=1;
(2)if (N>1) then
      {
        (2.1)do the following N-1 times:
            (2.1.1) fac=fac*N;
            (2.1.1) N--;

      }
(3)print fac;

(b)
fac(n):
(1) if (n=1 or n=0) then fac(n)=1
    else fac(n)=fac(n-1)*n;

main():
(1) fac(N);
```

题目 (DH: 2.8)

Show how to simulate a "while-do"loop by conditional statements and a recursive procedure.

解答:

```
while(A) do B;


rec():
(1) if (A) then
    {
```

```
      do B;
      rec();
    }
main():
(1) rec();
```

第二部分　订正

题目 (UD:4.5)

Negate the following sentences. If you don't know how to negate it,change it to symbols and then negate. State the universe, if appropriate.

(j) For all ε>0,there exist δ>0 such that if x is a real number with |x−1|<δ,then $|x^2 - 1| < \varepsilon$.

(k) For all real numbers M,there exists a real number N such that |f(n)|>M for all n > N.

订正:

(j) There exists a ε >0 ,for every δ >0 ,such that if x is a real number with |x-1|<δ,then $|x^2$ -1| $\geq \varepsilon$.

(k) 批改有误

题目 (UD:4.7)

Consider the following statement:

$\forall x, ((x \in Z \wedge \neg(\exists y, (y \in Z \wedge x = 7y))) \rightarrow (\exists z, (z \in Z \wedge x = 2z)))$.

(a) Negate this statement.

订正:

(a) 批改有误

第三部分　反馈

你可以写:

- 对课程及教师的建议与意见
- 教材中不理解的内容

- 希望深入了解的内容
- 等