

SỞ GD-ĐT KHÁNH HÒA

=====



TÀI LIỆU TẬP HUẤN CHUYÊN MÔN THCS
MÔN: TIN HỌC

CHUYÊN ĐỀ
KỸ THUẬT ĐỆ QUY
PHƯƠNG PHÁP QUAY LUI

ThS. Dương Khắc Hưởng
Trường THPT chuyên Lê Quý Đôn

Nha Trang, tháng 8/2018

MỤC LỤC

CHUYÊN ĐỀ 1: KỸ THUẬT ĐỆ QUY	2
1. Đệ quy là gì?	2
2. Bài toán đệ quy	3
3. Lập trình đệ quy	3
4. Một số ví dụ về cài đặt chương trình đệ quy	4
CHUYÊN ĐỀ 2: PHƯƠNG PHÁP QUAY LUI.....	8
1. Thế nào phương pháp quay lui?.....	8
2. Bài toán quay lui vết cạn tổng quát.....	8
3. Ý tưởng phương pháp quay lui	9
4. Một số bài toán ứng dụng phương pháp quay lui	9
BÀI TẬP	17
Bài 40	17
Bài 41:.....	17
Bài 42:.....	18
Bài 43:.....	18
Bài 44:.....	19
Bài 45:.....	19
Bài 46:.....	20
Bài 47:.....	20
Bài 48:.....	21
Bài 49:.....	21
TÀI LIỆU THAM KHẢO	23

CHUYÊN ĐỀ 1: KỸ THUẬT ĐỆ QUY

1. Đệ quy là gì?

Trong thực tế, một đối tượng được gọi là đệ quy (recursion) nếu nó được mô tả lại thông qua chính nó. Nghĩa là, các đối tượng này được định nghĩa một cách quy nạp từ những khái niệm đơn giản nhất cùng dạng với nó.

Ví dụ để định nghĩa tổ tiên của một người, người ta định nghĩa như sau:

** Bố mẹ của một người bất kỳ là tổ tiên của người đó (trường hợp cơ bản)*

** Bố mẹ của tổ tiên một người bất kỳ là tổ tiên của người đó (trường hợp đệ quy)*

Trong đời sống, ta cũng thường xuyên thấy một số hiện tượng đệ quy, ví dụ như hai chiếc gương đặt đối diện nhau, vòng xoắn ốc (trong vòng xoắn ốc có vòng xoắn ốc nhỏ hơn).

Trong toán học và khoa học máy tính, các tính chất (hoặc cấu trúc) được gọi là đệ quy nếu trong đó một lớp các đối tượng hoặc phương pháp được xác định bằng việc xác định một số rất ít các trường hợp hoặc phương pháp đơn giản (thông thường chỉ một) và sau đó xác định quy tắc đưa các trường hợp phức tạp về các trường hợp đơn giản.

Ví dụ trong toán học, một số bài toán được định nghĩa dạng đệ quy (quy nạp) như sau:

Định nghĩa số tự nhiên :

0 là một số tự nhiên

Nếu k là một số tự nhiên thì $k+1$ cũng là một số tự nhiên

Theo đó, ta sẽ có: $1 = 0 + 1$ là số tự nhiên, $2 = 1 + 1$ cũng là một số tự nhiên,...Cứ như vậy ta sẽ định nghĩa được các số tự nhiên khác lớn hơn. Do đó, số tự nhiên là khái niệm mang bản chất đệ quy.

Định nghĩa giai thừa của n ($n!$) :

Khi $n=0$, ta có $0!=1$

Khi $n>0$, ta có $n! = (n - 1)! \times n$

Như vậy, ta suy ra : $1! = 0! \times 1$, $2! = 1! \times 2$,...

Giai thừa cũng là một khái niệm mang tính đệ quy.

2. Bài toán đệ quy

Bài toán đệ quy là những bài toán mang bản chất đệ quy. Nghĩa là những bài toán này có thể được phân rã thành các bài toán nhỏ hơn, đơn giản hơn nhưng có cùng dạng với bài toán ban đầu. Những bài toán nhỏ lại được tiếp tục phân rã thành các bài toán nhỏ hơn. Cứ như vậy cho đến khi bài toán còn đơn giản đến mức có thể suy ra ngay kết quả mà không cần phải phân rã nữa (bài toán đã nhằm tính được kết quả).

Để giải các bài toán mang tính chất đệ quy ta phải giải tất cả các bài toán con rồi kết hợp các kết quả của các bài toán con đó lại để có được lời giải cho bài toán lớn ban đầu. Cách phân rã bài toán như vậy gọi là “chia để trị” (divide and conquer), là một dạng của đệ quy.

3. Lập trình đệ quy

Trong lập trình chúng ta có thể xây dựng các chương trình con đệ quy bằng cách đặt lời gọi đến chính nó trong thân của nó. Thông thường với chương trình con đệ quy khi được gọi ta phải truyền vào cho nó tham số và tham số của chương trình bị gọi phải nhỏ hơn tham số của chương trình con gọi nó. Việc làm như vậy dẫn đến chương trình được xử lý liên tục cho đến khi tham số truyền vào gặp điểm dừng (điểm neo) tại một vị trí đặc biệt được gọi là trường hợp suy biến (degenerate) hoặc chưa đến điểm neo mà bộ nhớ bị vét cạn (tràn) thì chương trình sẽ báo lỗi và dừng lại. Ví dụ trong ngôn ngữ lập trình Pascal sẽ báo lỗi “stack over flow”.

Trong lập trình, khi một chương trình con được gọi sẽ được đưa vào vùng nhớ mới trong bộ nhớ Stack, chương trình con đệ quy cũng vậy, mỗi lần có lệnh gọi, chương trình con đệ quy cũng sẽ được đưa vào Stack và nếu như không có điểm dừng, hoặc gọi mãi mà chưa tới điểm dừng chương trình sẽ gây ra lỗi tràn bộ nhớ.

Như vậy, việc xây dựng được một chương trình con đệ quy để giải một bài ta phải có những điều kiện sau:

- + Bài toán đó phải định nghĩa được dưới dạng đệ quy;
- + Trong định nghĩa phải có trường hợp suy biến;
- + Sau mỗi lần gọi đệ quy tham số của chương trình con phải nhỏ dần cho đến khi gặp trường hợp suy biến.

Lược đồ chung của một chương trình con đệ quy sẽ có dạng như sau:

Procedure/Function <Tên chương trình con> (<tham số>);

Begin

 If (trường hợp suy biến) then <xử lý suy biến>

 Else

 Begin

 [lệnh tiền đệ quy];

 <Xử lý trường hợp đệ quy>;

 [lệnh hậu đệ quy];

 End;

End;

4. Một số ví dụ về cài đặt chương trình đệ quy

Ví dụ 1: Bài toán tính giai thừa của số nguyên n.

Định nghĩa bài toán

$GT(n) = 1$ nếu $n=0$;

$GT(n) = n * GT(n-1)$ nếu $n>0$.

Cài đặt

Function GT(n:longint):longint;

Begin

 If n=0 then exit(1)

 Else exit(n*GT(n-1))

End;

Ví dụ 2: Bài toán tìm số thứ n trong dãy Fibonacci

Định nghĩa bài toán

$F(n) = 1$ nếu $n=1$ hoặc $n=2$

$F(n) = F(n-1)+F(n-2)$ nếu $n>2$

Cài đặt

Function F(n:longint):longint;

Begin

```
    If (n=1) or (n=2) then exit(1)
    Else exit(F(n-1)+F(n-2))
End;
```

Ví dụ 3: Tìm chuỗi đảo ngược của một chuỗi S có độ dài là n

Định nghĩa bài toán

$nguoc(S) = S$ nếu $n=1$
 $nguoc(S) = S[n] + nguoc(copy(S, 1, n-1))$;

Cài đặt

```
Function Nguoc(s:string):longint;
Var n:integer;
Begin
    n:=length(s);
    If n=1 then exit(S)
    Else exit(S[n]+Nguoc(copy(S,1,n-1)))
End;
```

Có thể cài đặt bằng thủ tục như sau

```
procedure Nguoc(s:string);
begin
    if length(s)=1 then write(s)
    else
    begin
        write(S[length(s)]);
        Nguoc(copy(S,1,length(s)-1));
    end;
end;
```

Ví dụ 4: Tìm ước chung lớn nhất của hai số nguyên dương a và b

Định nghĩa bài toán.

$UCLN(a,b) = a$ nếu $a = b$
 $UCLN(a,b) = UCLN(\max(a,b) - \min(a,b), \min(a,b))$

Cài đặt

```
Function ucln(a,b:longint):longint;  
Begin  
    If a=b then exit(a)  
    Else exit(UCLN(max(a,b)-min(a,b),min(a,b)))  
End;
```

Ví dụ 5: Bài toán tìm số lớn nhất trong mảng A có n phần tử

Định nghĩa bài toán

$A_{\max}(A,n) = A[1]$ nếu $n=1$;
 $A_{\max}(A,n) = \max(A_{\max}(A,n-1), A[n])$;

Cài đặt

```
uses math;  
const maxn=1000;  
type arr = array[1..maxn] of longint;  
var A:arr;  
    n,i:longint;  
function Amax(A:Arr;n:longint):longint;  
begin  
    if n=1 then exit(A[1])  
    else exit(max(A[n],Amax(A,n-1)));  
end;  
begin  
    write('n = ');  
    readln(n);  
    for i:=1 to n do  
    begin  
        write('A[' ,i ,']= ');  
        readln(A[i]);  
    end;  
    write('Max = ',Amax(A,n));  
    readln;  
End.
```

Ví dụ 6: Tính tổng các chữ số của một số nguyên dương

Định nghĩa bài toán

$$\text{Tongcs}(n) = n \text{ nếu } n \leq 9$$

$$\text{Tongcs}(n) = n \bmod 10 + \text{Tongcs}(n \div 10) \text{ nếu } n > 9$$

Cài đặt

```
Function tongcs(n:longint):longint;  
Begin  
    If n<=9 then exit(n)  
    Else exit(n mod 10 + tongcs(n div 10));  
End;
```


CHUYÊN ĐỀ 2: PHƯƠNG PHÁP QUAY LUI

1. Thế nào phương pháp quay lui?

Quay lui, vét cạn hay duyệt toàn bộ... là tên gọi tuy không đồng nghĩa nhưng cùng chỉ một phương pháp tìm nghiệm của bài toán trong tin học. *Phương pháp quay lui là phương pháp tìm nghiệm của bài toán bằng cách xét hết tất cả các khả năng có thể.* Đối với con người phương pháp này thường là không khả thi vì số phương án cần kiểm tra lớn. Tuy nhiên đối với máy tính, nhờ tốc độ xử lý nhanh, máy tính có thể giải rất nhiều bài toán bằng phương pháp quay, lui vét cạn.

Ưu điểm của phương pháp quay lui, vét cạn là luôn đảm bảo tìm ra nghiệm đúng, chính xác. Tuy nhiên, hạn chế của phương pháp này là thời gian thực thi lâu, độ phức tạp lớn. Do đó vét cạn thường chỉ phù hợp với các bài toán có kích thước nhỏ.

2. Bài toán quay lui vét cạn tổng quát

Bài toán dạng quay lui vét cạn thường được phát biểu một cách tổng quát như sau: Cần tìm một cấu hình được mô tả bởi một bộ phận gồm n thành phần $X = \{x_1, x_2, \dots, x_n\}$. Mỗi thành phần x_i ($i = 1..n$) có m khả năng chọn từ một tập $S = \{s_1, s_2, \dots, s_m\}$ thỏa mãn điều kiện nào đó.

Với bài toán trên, chúng ta cần tìm những thành phần sau:

- $X = \{x_1, x_2, \dots, x_n\}$, được gọi là vector nghiệm
- $S = \{s_1, s_2, \dots, s_m\}$, được gọi là tập ứng cử viên
- Điều kiện ràng buộc của vector nghiệm.

Tùy theo từng bài toán mà vector nghiệm có thể là nhiệm của bài toán hoặc có thể là một vector cấu hình để từ đó chúng ta có thể tìm ra nghiệm của bài toán. Đối với điều kiện ràng buộc của vector nghiệm cũng tùy theo từng bài toán mà chúng có thể khác nhau. Điều kiện ràng buộc của vector nghiệm có thể liên quan đến từng thành phần trong vector nghiệm hoặc liên quan đến toàn thể cấu hình vector nghiệm.

3. Ý tưởng phương pháp quay lui

Giả sử chúng ta đã tìm được $i-1$ thành phần cho vector nghiệm x_1, x_2, \dots, x_{i-1} . Để tìm tiếp thành phần thứ i tức là x_i , ta có m khả năng chọn từ tập $S = \{s_1, s_2, \dots, s_m\}$. Trong trường hợp này có hai khả năng xảy ra:

+ Nếu tìm được s_j ($j = 1..m$) cho x_i thì chọn s_j cho x_i và nếu đủ cấu hình nghiệm ($i=n$) thì ghi nhận một cấu hình nghiệm, ngược lại tiếp tục chọn cho thành phần thứ $i+1$.

+ Nếu thử hết tất cả m khả năng mà không có một khả năng nào thỏa mãn cho x_i thì quay lui chọn lại cho thành phần thứ $i-1$.

Lược đồ chung phương pháp quay lui được mô tả như sau:

```
Procedure Try(i)
Begin
  For j:=1 to m do
  Begin
    If <thỏa ĐK> then
    Begin
      X[i]:=S[j];
      If <đủ cấu hình nghiệm> then GhiNghiem
      Else Try(i+1);
      <Hủy bước chọn thứ i>
    End;
  End;
End;
```

4. Một số bài toán ứng dụng phương pháp quay lui

Bài 1: Cho dãy số nguyên dương A_1, A_2, \dots, A_n và số nguyên dương P . Hãy chọn một số các số trong dãy sao cho tổng của các số đã chọn lớn nhất và không vượt quá P .

Dữ liệu vào: tệp văn bản VD1.INP gồm:

- + Dòng đầu ghi hai số nguyên dương n và P ($2 \leq n \leq 20$; $P \leq 1000$);
- + Dòng thứ hai ghi n số trong dãy A , giữa các số cách nhau một dấu cách và có giá trị không vượt quá 100.

Kết quả: Ghi ra tệp văn bản VD1.OUT gồm:

- + Dòng đầu ghi tổng các số được chọn;
- + Dòng thứ hai ghi chỉ số của các số được chọn.

Ví dụ

BAI1.INP	BAI1.OUT
10 16 1 4 7 6 5 3 9 10 2 4	16 8 9 10

Phân tích bài toán

+ Vector nghiệm $X = \{x_1, x_2, \dots, x_n\}$, trong đó nếu $x_i = 1$ thì A_i được chọn, còn nếu $x_i = 0$ thì A_i không được chọn.

+ Tập ứng cử viên $S = \{0, 1\}$

+ Điều kiện: $\sum_1^n x_i A_i \leq P$ và đạt Max

Cài đặt

```
const    fi='bai1.inp';
         fo='bai1.out';
         nmax=50;
var      A:array[1..nmax] of longint;
         X,B:array[1..nmax] of 0..1;
         n,p,v,vmax:longint;

procedure ghinghiem;
var v,i:longint;
begin
    v:=0;
    for i:=1 to n do
        if X[i]=1 then v:=v+A[i];
    if (v>vmax) and (v<=p) then
        begin
            vmax:=v;
            move(X,B,sizeof(X));
        end;
end;
procedure try(i:longint);
var j:0..1;
begin
    for j:=0 to 1 do
```

```
begin
    X[i]:=j;
    if i=n then ghinghiem
    else try(i+1);
end;
end;
procedure Nhap;
var f:text;
    i:longint;
begin
    assign(f,fi);
    reset(f);
    readln(f,n,p);
    for i:=1 to n do read(f,A[i]);
    close(f);
end;
procedure Xuat;
var f:text;
    i:longint;
begin
    assign(f,fo);
    rewrite(f);
    writeln(f,vmax);
    for i:=1 to n do
        if B[i]=1 then write(f,i,' ');
    close(f);
end;
Begin
    Nhap;
    Try(1);
    Xuat;
End.
```

Bài 2: Hãy liệt kê tất cả các xâu ký tự khác nhau có độ dài n được thành lập từ ba chữ cái ‘A’, ‘B’ và ‘C’ sao cho không có hai ký tự giống nhau đứng cạnh nhau và mỗi ký tự phải xuất hiện ít nhất một lần.

Dữ liệu vào: Tập văn bản BAI2.INP chỉ ghi số nguyên n ($3 \leq n \leq 20$).

Kết quả: Tập văn bản BAI2.OUT tất cả các xâu ký tự tìm được, mỗi xâu trên một dòng.

Ví dụ:

BAI2.INP	BAI2.OUT
3	ABC ACB BAC BCA CAB CBA

Phân tích bài toán

- + Vector nghiệm $X = \{x_1, x_2, \dots, x_n\}$
- + Tập ứng cử viên $S = \{ 'A', 'B', 'C' \}$
- + Điều kiện: $x_i \neq x_{i-1}$ ($i=2..n$) và 'A', 'B' và 'C' phải có mặt trong X

Cài đặt

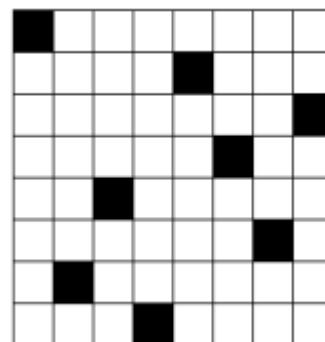
```
const      fi='bai2.inp';
           fo='bai2.out';
           nmax=50;
var        X:string;
           n,i:longint;
           f:text;
procedure ghinghiem;
var v,i:longint;
begin
    if (pos('A',X)>0) and (pos('B',X)>0) and (pos('C',X)>0)
    then writeln(f,X);
end;
procedure try(i:longint);
var j:char;
begin
    for j:='A' to 'C' do
    begin
        if j<>X[i-1] then
        begin
            X[i]:=j;
            if i=n then ghinghiem
            else try(i+1);
        end;
    end;
end;
procedure Nhap;
var f:text;
    i:longint;
begin
```

```
    assign(f, fi);  
    reset(f);  
    readln(f, n);  
    close(f);  
end;  
Begin  
    Nhap;  
    X:='';  
    for i:=1 to n do X:=X+'A';  
    assign(f, fo);  
    rewrite(f);  
    Try(1);  
    close(f);  
end.
```

Bài 3: Bài toán 8 quân hậu

Cần đặt 8 quân hậu trên bàn cờ vua 8 x 8 sao cho chúng không tấn công nhau, tức là không có hai quân hậu nào cùng nằm trên một hàng, một cột và một đường chéo.

Ví dụ: Hình bên là một cách đặt hậu thỏa mãn yêu cầu bài toán, các ô tô màu là vị trí đặt hậu.



Phân tích bài toán

Mỗi quân hậu sẽ được đặt trên một hàng nên chúng ta chỉ cần tìm cột chưa được đặt để đặt từng quân hậu. Quân hậu thứ i ($i=1..8$) sẽ được đặt trên cột x_i cho nên vector nghiệm của bài toán là $X = \{x_1, x_2, \dots, x_8\}$

Giá trị của x_i sẽ được chọn trong tập $\{1, 2, 3, 4, 5, 6, 7, 8\}$ nên tập ứng cử viên cho vector nghiệm là $S = \{1, 2, 3, 4, 5, 6, 7, 8\}$.

Điều kiện của vector nghiệm sẽ là:

- + Không có hai quân hậu cùng cột, nghĩa là với mọi $i \neq j$ thì $x_i \neq x_j$
- + Dễ dàng nhận thấy nếu hai ô (x_1, y_1) và (x_2, y_2) cùng nằm trên đường chéo chính (từ trên phải xuống dưới trái) thì $x_1 - y_1 = x_2 - y_2$. Và nếu hai ô (x_1, y_1) và (x_2, y_2) cùng nằm trên đường chéo phụ (từ trên trái xuống dưới phải) thì $x_1 + y_1 = x_2 + y_2$ cho nên điều kiện để quân hậu thứ i và thứ j ($i \neq j$) không đặt trên cùng một đường chéo sẽ là: $i - x_i \neq j - x_j$ và $i + x_i \neq j + x_j$.

Từ những phân tích trên, chúng ta có thể xây dựng cấu trúc dữ liệu cho chương trình như sau:

- + Mảng X[1..8] dùng để lưu trữ vector nghiệm.
- + Mảng C[1..8] dùng để đánh dấu cột đã được đặt quân hậu hay chưa.
- + Mảng CC[-7..7] dùng để đánh dấu đường chéo chính đã được đặt quân hậu hay chưa.
- + Mảng CP[2..16] dùng để đánh dấu đường chéo phụ đã được đặt quân hậu hay chưa.

Cài đặt chương trình

```
const      maxn = 20;

           fi = 'bai3.inp';
           fo = 'bai3.out';

var  B:array[1..maxn,1..maxn] of char;
     CC:array[-maxn..maxn] of boolean;
     CP:array[2..maxn *2] of boolean;
     C:array[1..maxn] of boolean;
     X:array[1..maxn] of integer;
     n,dem:integer;
     f:text;

procedure Input;
var f:text;
begin
    assign(f,fi);
    reset(f);
    readln(f,n);
    close(f);
end;

procedure ghinghiem;
var i,j:integer;
begin
    inc(dem);
```

```
        Writeln(f, 'Cach thu: ', dem);
        for i:=1 to n do write(f, X[i]:5);
        writeln(f);
end;
procedure tryby(i:integer);
var j:integer;
begin
    for j:=1 to n do
    begin
        if not(C[j]) and not(CC[i-j]) and not(CP[i+j]) then
        begin
            X[i]:=j;
            C[j]:=true;
            CC[i-j]:=true;
            CP[i+j]:=true;
            if i=n then ghinghiem
            else tryby(i+1);
            C[j]:=false;
            CC[i-j]:=false;
            CP[i+j]:=false;
        end;
    end;
end;
BEGIN
    Input;
    fillchar(C, sizeof(C), false);
    fillchar(CP, sizeof(CP), false);
    fillchar(CC, sizeof(CC), false);
    dem:=0;
    assign(f, fo); rewrite(f);
```



```
tryby(1);  
close(f);  
END.
```

BÀI TẬP

Bài 40

Số siêu nguyên tố là số nguyên tố mà khi bỏ một số tùy ý các chữ số bên phải của nó thì phần còn lại vẫn là số nguyên tố. Ví dụ số 2333 là số siêu nguyên tố có 4 chữ số vì các số 233, 23, 2 đều là số nguyên tố.

Cho số n ($1 < n < 10$), hãy đưa ra các số nguyên tố có n chữ số.

Dữ liệu vào: Từ tệp văn bản BAI40.INP chỉ ghi số nguyên dương n .

Kết quả: Ghi ra tệp văn bản BAI40.OUT các số siêu nguyên tố có n chữ số, mỗi số trên một dòng và theo thứ tự từ nhỏ đến lớn.

Ví dụ

BAI40.INP	BAI40.OUT
4	2333 2339 2393 2399 2939 3119 3137 3733 3739 3793 3797 5939 7193 7331 7333 7393

Bài 41:

Cho số nguyên dương N ($N \leq 10^9$). Hãy tìm tất cả các hoán vị của các chữ số của N . Lưu ý, chữ số 0 đứng đầu không có nghĩa.

Dữ liệu vào: Tệp văn bản BAI41.INP chỉ ghi số nguyên N .

Kết quả: Tệp văn bản BAI41.OUT ghi các hoán vị của các chữ số của N theo thứ tự tăng dần. Mỗi hoán vị được ghi trên một dòng.

Ví dụ

BAI41.INP	BAI41.OUT
1223	1223 1232 1322 2123 2132 2213 2231 2312 2321 3122 3212 3221

Bài 42:

Hãy liệt kê dãy nhị phân có độ dài n ($n \leq 10$) theo thứ tự từ điển

Dữ liệu vào: Từ tệp văn bản BAI42.INP chỉ ghi số nguyên dương n .

Kết quả: Đưa ra tệp văn bản BAI42.OUT, mỗi dòng là một dãy nhị phân theo yêu cầu bài toán.

Ví dụ

BAI42.INP	BAI42.OUT
3	000 001 010 011 100 101 110 111

Bài 43:

Cho xâu S (độ dài không vượt quá 10) chỉ gồm các ký tự từ 'A' đến 'Z' (các ký tự trong xâu S đôi một khác nhau). Hãy liệt kê tất cả các hoán vị khác nhau của các ký tự trong xâu S .

Dữ liệu vào: Từ tệp văn bản BAI43.INP ghi xâu S

Kết quả: Ghi ra tệp văn bản BAI43.out ghi mỗi hoán vị của xâu S trên mỗi dòng.

Ví dụ

BAI43.INP	BAI43.OUT
XYZ	XYZ

	XZY YXZ YZX ZXY ZYX
--	---------------------------------

Bài 44:

Cho số nguyên dương n ($n \leq 20$). Hãy liệt kê tất cả các xâu có độ dài n chỉ chứa hai ký tự ‘A’ hoặc ‘B’ mà trong xâu không có hai ký tự ‘B’ đứng liền nhau.

Dữ liệu vào: Từ tệp văn bản BAI44.INP chỉ ghi số nguyên dương n .

Kết quả: Ghi ra tệp văn bản BAI44.OUT, mỗi dòng một xâu ký tự thỏa mãn điều kiện bào toán.

Ví dụ

BAI44.INP	BAI44.OUT
3	AAA AAB ABA BAA BAB

Bài 45:

Cho dãy số A gồm n số ($1 < n \leq 10$) a_1, a_2, \dots, a_n và một số nguyên dương K ($1 < K \leq n/2$). Hãy tìm cách chia dãy số A thành K nhóm sao cho tổng của các nhóm bằng nhau.

Dữ liệu vào: Từ tệp văn bản BAI45.inp gồm:

- + Dòng đầu ghi hai số nguyên n và K cách nhau một dấu cách.
- + Dòng thứ hai ghi n số của dãy A , mỗi số không vượt quá 100, giữa các số cách nhau một dấu cách.

Kết quả: Ghi ra tệp văn bản BAI45.OUT gồm K dòng, mỗi dòng là các số thuộc cùng một nhóm. Nếu không chia được thì ghi số -1.

Ví dụ

BAI45.INP	BAI45.OUT
5 3 1 4 6 9 10	1 9 4 6 10

Bài 46:

Một chuỗi $X = x_1x_2\dots x_M$ được gọi là một chuỗi con của chuỗi $Y = y_1y_2\dots y_N$ nếu ta có thể nhận được chuỗi X từ chuỗi Y bằng cách xóa đi một số ký tự trong chuỗi Y (các ký tự còn lại giữ nguyên thứ tự). Ví dụ chuỗi ‘adz’ là chuỗi con của chuỗi ‘bcadazdzab’.

Yêu cầu: Cho chuỗi ký tự S chỉ gồm các ký tự từ ‘a’ đến ‘z’. Hãy tìm tất cả các chuỗi con của S .

Dữ liệu vào: Từ tệp văn bản BAI46.INP chỉ ghi chuỗi S có độ dài không quá 15.

Kết quả: Ghi ra tệp văn bản BAI46.OUT các chuỗi con của chuỗi S , mỗi chuỗi trên một dòng
Ví dụ

BAI46.INP	BAI46.OUT
aba	a b ba aa ab aba

Bài 47:

Cho số nguyên dương n ($n \leq 10$). Hãy liệt kê tất cả các cách đặt n dấu ngoặc mở và n dấu ngoặc đóng đúng theo quy tắc ưu tiên tính toán giá trị một biểu thức toán học.

Dữ liệu vào: Từ tệp văn bản BAI47.INP chỉ ghi số nguyên dương n .

Kết quả: Ghi ra tệp văn bản BAI47.OUT trên mỗi dòng là một chuỗi dấu ngoặc thể hiện một cách đặt tìm được.

Ví dụ:

BAI47.INP	BAI47.OUT
3	((())) (())() (())() ()() ()()

Bài 48:

Cho n số nguyên dương ($n \leq 10$) a_1, a_2, \dots, a_n ($a_i \leq 100$). Hãy tìm số nguyên dương m nhỏ nhất sao cho m không phân tích được tổng của một số các số (mỗi số chỉ sử dụng một lần) trong n số trên.

Dữ liệu vào: Tập văn bản BAI48.INP gồm:

- + Dòng đầu ghi số nguyên dương n
- + Dòng thứ hai ghi n số nguyên dương a_1, a_2, \dots, a_n , giữa các số cách nhau một dấu cách.

Kết quả: Ghi ra tập văn bản BAI48.OUT số m tìm được.

Ví dụ

BAI48.INP	BAI48.OUT
4 1 2 3 6	13

Bài 49:

Cho chuỗi S (độ dài không vượt quá 10) chỉ gồm các ký tự từ 'A' đến 'Z'. Các ký tự trong chuỗi không nhất thiết phải khác nhau. Hãy liệt kê tất cả các hoán vị khác nhau của chuỗi S .

Dữ liệu vào: Từ tập văn bản BAI49.INP chỉ ghi chuỗi S .

Kết quả: Ghi ra tập văn bản BAI49.OUT tất cả các hoán vị của chuỗi S . Mỗi hoán vị được ghi trên một dòng.

Ví dụ

BAI49.INP	BAI49.OUT
AAB	AAB ABA BAA

TÀI LIỆU THAM KHẢO

[1] *Hồ Sỹ Đàm, Đỗ Đức Đông, Lê Minh Hoàng, Nguyễn Thanh Hùng* - Tài liệu giáo khoa chuyên tin quyển 1 – Nhà xuất bản giáo dục.