



DẶNG TUẤN THÀNH

(Sưu tầm và biên soạn)



KỸ THUẬT TỔ CHỨC CHƯƠNG TRÌNH

```
each: function(e, t, n) {
    var r, i = 0;
    o = e.length,
    a = M(e);
    if (n) {
        if (a) {
            for (; o > i; i++)
                if (r = t.apply(e[i], n), r === !1) break;
        } else
            for (i in e)
                if (r = t.apply(e[i], n), r === !1) break;
    } else if (a) {
        for (; o > i; i++)
            if (r = t.call(e[i], i, e[i]), r === !1) break;
    } else
        for (i in e)
            if (r = t.call(e[i], i, e[i]), r === !1) break;
    return e
},
trim: b && b.call("\uffeff\u00a0") ? function(e) {
    return null == e ? "" : b.call(e)
} : function(e) {
    return null == e ? "" : (e + "").replace(c, "")
},
makeArray: function(e, t) {
    var n = t || [];
    return null != e && (M(Object(e)) ? x.merge(n, "string" == typeof e ? [e] : e) : h.call(n, e)), n
},
isArray: function(e, t, n) {
    var r;
    if (t) {
        if (n) return h.call(t, e, n);
        for (r = t.length, n = n ? 0 > n ? Math.max(0, r + n) : n : 0; r > n; n++)
            if (n in t && t[n] === e) return n
    }
}
```





“Hãy duy trì một thái độ đúng đắn, một tinh thần dũng cảm, chân thành và vui vẻ. Mọi cơ hội sẽ đến từ sự khát khao và mọi mong ước chân thành đều sẽ được đáp ứng. Hãy ngẩng đầu thật cao và hiên ngang bước tới. Tất cả chúng ta đều là những nhân tài tiềm ẩn trong tư chất của chính mình.”

Elbert Green Hubbard



LỜI MỞ ĐẦU



Nhằm cung cấp tài liệu, trang bị kỹ năng tự học, tự nghiên cứu cho học sinh và giáo viên, hình thành và phát triển kỹ năng cho người học; Xây dựng hệ thống bài tập củng cố kiến thức bộ môn theo chương trình môn Tin học; Bổ sung kiến thức nâng cao, bồi dưỡng học sinh giỏi môn Tin học THCS và THPT, tác giả sưu tầm và biên soạn cuốn tài liệu: “**Kỹ thuật tổ chức chương trình**”.

Giả mã và **code chương trình** trong “Kỹ thuật tổ chức chương trình” được viết bởi ngôn ngữ lập trình C++ và Pascal. Nội dung gồm **71** bài tập của các chuyên đề được biên soạn kèm **video bài giảng** chi tiết tạo điều kiện thuận lợi cho việc tự học, tự nghiên cứu. Với mỗi đơn vị kiến thức, có các bài tập, hướng dẫn thuật toán và code chương trình giúp người học nhanh chóng hình thành kỹ thuật tổ chức chương trình; Người học được hướng dẫn thuật toán có mức độ hoàn thiện bài khác nhau, phát triển kỹ năng làm mịn chương trình.

Trong thời gian tới, tác giả viết tiếp tài liệu mới “**21 giờ học quy hoạch động**” với các chuyên đề liên quan đến kỳ thi học sinh giỏi các cấp. Rất mong các bạn đón đọc.

Tài liệu này được biên soạn rất tỉ mỉ nhưng không tránh được những thiếu sót, tác giả rất mong được sự đóng góp nội dung, phản hồi của bạn đọc vào mail: dtthanh.c3ntt@yenbai.edu.vn.

Tác giả

Đặng Tuấn Thành

Giáo viên Tin học

Trường THPT Chuyên Nguyễn Tất Thành, tỉnh Yên Bái

SĐT: 0854518333/0366996363



Mục lục

KỸ THUẬT TỐ CHỨC CHƯƠNG TRÌNH.....	6
1) DÃY SỐ WAVIO	6
2) TẦN SỐ	7
3) DÃY CON	9
4) THÀNH LUÝ	10
5) TỔNG 4 SỐ	12
6) TỔNG 4 SỐ NGUYÊN TỐ	13
7) TÌM ƯỚC GCD.*	14
8) SỐ NGHỊCH THẾ.....	15
9) SỐ ĐẸP.....	16
10) SỐ ĐẸP 2.....	20
11) TÍCH CÁC CHỮ SỐ.....	22
12) PHƯƠNG TRÌNH.....	24
13) DÃY SỐ.....	24
14) MÔ ĐUN M.....	26
15) ĐẦU TU CHÚNG KHOÁN	28
16) CHUỖI NHỊ PHÂN	31
17) LŨY THỪA 2	33
18) CÂY SỐI	35
19) CHỮ NHẬT	37
20) SỐ LUỢNG	37
21) LỚP HỌC MÚA	37
22) DÃY SỐ HAMMING	38
23) TỔNG NGUYÊN TỐ.....	41
24) XÂU CON	43
25) Ô VUÔNG	44
26) TỔNG CẶP SỐ	46
27) XẾP SÁCH	51
28) NHÓM 3	51
29) GIẢI MÃ	52
30) DÃY SỐ ĐẶC BIỆT	54
31) XÂU FIBONACCI.....	57
32) MUỜI MỘT	57
33) CHỮ SỐ HÀNG ĐƠN VỊ	61
34) SỐ ĐỒI XỨNG	62
35) GIẢI MÃ SỐ.....	65
36) SỐ MỘT	67
37) CÁC NỀN VĂN MINH CỔ ĐẠI	68
38) TỔNG SỐ NGUYÊN TỐ 1	69
39) TỔNG SỐ NGUYÊN TỐ 2	74
40) CHỌN QUÀ	79
41) QLHVHT.*	80
42) DQHV6.*	80
43) DQHV7.*	81



44)	BỘI SỐ CHUNG NHỎ NHẤT.....	81
45)	SỐ NHỊ THẾ.....	83
46)	THÚ TỰ TỰ ĐIỀN CÓ TRỌNG SỐ.....	86
47)	ĐẾM CHỮ SỐ.....	88
48)	SỐ ĐẸP.....	90
49)	ƯỚC SỐ CHUNG LỚN NHẤT	96
50)	XÂU CON	100
51)	PALINDROME.....	103
52)	THU GỌN.....	104
53)	TAM GIÁC PASCAL	106
54)	TỔNG	108
55)	TỔNG HAI SỐ	109
56)	GIÁ TRỊ GẦN NHẤT	111
57)	ĐẾM SỐ LUỢNG.....	112
58)	ƯỚC SỐ.....	114
59)	KHÔNG VÀ MỘT	116
60)	ĐAHOÁN VỊ NGUYÊN TỐ.....	117
61)	QUÂN XE.....	123
62)	CÂN ĐĨA.....	123
63)	TẬP SỐ.....	123
64)	SỐ RỖ RÀNG.....	125
65)	DÃY SỐ.....	128
66)	TAM GIÁC.....	130
67)	CẶP SỐ 0.....	131
68)	GHÉP SỐ.....	131
69)	SỐ NGUỒN	132
70)	SỐ PINARY.....	132
71)	CHỌN HÌNH	133



KỸ THUẬT TỐ CHỨC CHƯƠNG TRÌNH



1) DÃY SỐ WAVIO

Dãy số Wavio là dãy số nguyên thoả mãn các tính chất:

- Số lượng phần tử (độ dài của dãy) L là lẻ, tức là $L = 2*N+1$,
- $N+1$ phần tử đầu tiên là một dãy tăng ngặt,
- $N+1$ phần tử cuối là một dãy giảm ngặt.

Ví dụ, dãy số **1, 2, 3, 4, 5, 4, 3, 2, 0** là một dãy Wavio độ dài 9, còn dãy **1, 2, 3, 4, 5, 4, 3, 2, 2** không phải là dãy Wavio.

Yêu cầu: Cho dãy số nguyên độ dài N, hãy tìm dãy con Wavio độ dài lớn nhất từ dãy đã cho.

Ví dụ: cho dãy **1 2 3 2 1 2 3 4 3 2 1 5 4 1 2 3 2 2 1**. Dãy con Wavio dài nhất là **1 2 3 4 5 4 3 2 1** và có độ dài 9.

Dữ liệu: Vào từ file văn bản WAVIO.INP gồm:

- Dòng thứ nhất chứa số nguyên N ($0 < N \leq 10\,000$),
- Dòng thứ 2 chứa N số nguyên, các số cách nhau một dấu cách.

Kết quả: Đưa ra file văn bản WAVIO.OUT độ dài của dãy con Wavio dài nhất. **Ví dụ:**

WAVIO.INP	WAVIO.OUT
10	9
1 2 3 4 5 4 3 2 1 10	

Thuật toán:

<https://youtu.be/AqnGy2xDgAE>

- Định nghĩa dãy Wavio là dãy có nửa đầu tăng ngặt., nửa sau là dãy giảm ngặt và số lượng phần tử mỗi nửa bằng nhau.
- Sinh dãy nhị phân có độ dài N lưu vào mảng b.
- Với mỗi cấu hình nếu $b[i] = 1$ thì ta đẩy $a[i]$ vào mảng C có K phần tử.
- Kiểm tra K có thoả mãn là dãy Wavio hay không? Nếu có cập nhật Res.

```

Procedure xuli();
Var i, k, p: longint;
Begin
    K:=0;
    For i:=1 to n do
        If b[i]= 1 then
            Begin
                Inc(k);
                C[k]:= a[i];
            End;
    P:= 1;
    For i:=1 to k div 2 do
        If c[i] >= c[i+1] then
            Begin
                P:=0; break;
            End;
    For i:= k div 2 to k do
        If c[i] < c[i+1] then

```



```
Begin
    P:=0; break;
End;
If p = 1 then res := max(res,k);
End;
```

```
Free Pascal IDE
File Edit Search Run Compile Debug Tools Options Window Help
-[ ]-
var q,p,d:longint;
begin
    k:=0;
    for q:=1 to n do
        if b[q]=1 then
            begin
                inc(k);
                c[k]:=a[q];
            end;
    p:=1;
    if k mod 2=0 then d:=k div 2 else d:=k div 2 +1;
    for q:=1 to d-1 do
        if c[q]>=c[q+1] then
            begin
                p:=0;
                break;
            end;
    for q:= d to k-1 do
        if c[q]<=c[q+1] then
            begin
                p:=0;
                break;
            end;
    if p=1 then res:=max(res,k);
end;
procedure thu(i:longint);
var j:longint;
begin
    for j:=0 to 1 do
    begin
        b[i]:=j;
        if i=n then xuli
        else thu(i+1);
    end;
end;
BEGIN
    assign(input,'WAVIO.INP');reset(input);
    assign(output,'WAVIO.OUT');rewrite(output);
    readln(n);
    for i:=1 to n do read(a[i]);
    res:=0;
    thu(1);
    write(res);
    close(input);close(output);
END.
```

2) TẦN SỐ

Nhiều hằng cơ bản là số vô tỷ và các chữ số trong hằng xuất hiện gần ngẫu nhiên. Để kiểm tra xem các chữ số trong một hằng có thể sử dụng như bộ số ngẫu nhiên hay không người ta lấy chuỗi N chữ số liên tiếp của nó và tính tần số xuất hiện của các chuỗi số S_1, S_2, \dots, S_K ($1 \leq N \leq 1\ 000\ 000$, $1 \leq K \leq 1000$). Chuỗi S_i có độ dài không vượt quá 100.

Yêu cầu: Tính tần số xuất hiện các xâu S_i , $i = 1, 2, \dots, K$.

Dữ liệu: Vào từ file văn bản FREQ.INP:



- Dòng đầu tiên chứa số nguyên K – số lượng xâu
- Dòng thứ i trong K dòng tiếp theo chứa xâu ký tự số Si,
- Xâu S0.

Kết quả: Đưa ra file văn bản FREQ.OUT K số nguyên, mỗi số trên một dòng. Số nguyên thứ i xác định số lần xuất hiện Si trong xâu S0.

Ví dụ:

FREQ.INP	FREG.OUT
3	2
1828	1
90	0
1234	
2718281828459045235360	

Thuật toán:

<https://youtu.be/XbGpZcf257E>

- Đếm số lần xuất hiện S1 trong xâu S2: Tạo chức function dem(u,v: string): longint;
- Đi từ đầu xâu đến cuối xâu, tìm vị trí đầu tiên của u trong v. Nếu có vị trí này thì tăng res, xoá từ vị trí 1 đến vị trí tìm thấy. Lặp lại đến khi không còn xâu u trong xâu v.
- Kết quả của hàm là res.

```
Function dem(u,v: string) : longint;
Begin
  Res := 0;
  While pos(u, v) > 0 do
    Begin
      Inc(res);
      Delete(v, 1, pos(u,v));
    End;
    Exit(res);
  End;
```

- Chương trình chính:

```
Nhập N;
For(i,1,N) readln(a[i]);           //array[1..1000000] of string;
Readln(s);
For (i,1,N)
  Writeln(dem(a[i],s));
```



Free Pascal IDE



```

File Edit Search Run Compile Debug Tools Options Window
[ ]
var
  n,i:longint;
  a:array[1..1000000] of string;
  s:string;
function dem(u,v:string):longint;
var
  res:longint;
begin
  res:=0;
  while pos(u,v)>0 do
  begin
    inc(res);
    delete(v,pos(u,v),1);
  end;
  exit(res);
end;
BEGIN
  assign(input,'FREQ.INP');reset(input);
  assign(output,'FREQ.OUT');rewrite(output);
  readln(n);
  for i:=1 to n do readln(a[i]);
  readln(s);
  for i:=1 to n do writeln(dem(a[i],s));
  close(input);close(output);
END.

```

3) DÃY CON

Cho dãy N số nguyên ($1 \leq N \leq 10\,000$) A_1, A_2, \dots, A_N . Hãy tìm đoạn dài nhất các phần tử liên tiếp nhau cùng chia hết cho một số nguyên khác 1.

Dữ liệu: Vào từ file văn bản SUBSEQ.INP:

- Dòng đầu tiên chứa số nguyên N,
- Các dòng sau: chứa các số nguyên A_i , các số cách nhau ít nhất một dấu cách hoặc nhóm dấu xuống dòng.

Kết quả: Đưa ra file văn bản SUBSEQ.OUT một số nguyên xác định độ dài lớn nhất của dãy con tìm được.

Ví dụ:

SUBSEQ.INP	SUBSEQ.OUT
3	2
6 10 15	

Thuật toán:

<https://youtu.be/W8-FDerXzLo>

- Số nguyên dương khác một mà dãy một dãy con chia hết bằng ước chung lớn nhất của dãy con đó.
- Tô chúc hàm tìm UCLN(u,v) - Oclit
- For($i, 1, N-1$)
 - o For($j, i+1, N$)
 - Begin
 - P:=a[i];



For (k, i, j) p:= ucln(p, a[k]);
//P là ước chung lớn nhất của đoạn [i,j]
If p > 1 then res := max(res, j - i + 1);
End;
- Đưa res ra.

```
Free Pascal IDE
File Edit Search Run Compile Debug Tools Options Window Help
[ ]
uses math;
var
  n,i,j,k,res,p:longint;
  a:array[-1..1000001] of longint;
function ucln(u,v:longint):longint;
var
  r:longint;
begin
  while v>0 do
  begin
    r:=u mod v;
    u:=v;
    v:=r;
  end;
  exit(u);
end;
BEGIN
  assign(input,'SUBSEQ.INP');reset(input);
  assign(output,'SUBSEQ.OUT');rewrite(output);
  readln(n);
  for i:=1 to n do read(a[i]);
  for i:=1 to n-1 do
    for j:=i+1 to n do
    begin
      p:=a[i];
      for k:=i to j do p:=ucln(p,a[k]);
      if p>1 then res:=max(res,j-i+1);
    end;
  write(res);
  close(input);close(output);
END.
```

4) THÀNH LUÝ

Để đảm bảo an ninh chống lại sự tấn công của các bộ tộc khác tù trưởng xưa Fladland quyết định cho xây dựng các thành luỹ quanh các điểm dân cư đông đúc. Theo lời khuyên của thầy phù thuỷ, tên của các thành luỹ phải được chọn là một xâu con các ký tự liên tiếp nhau của tên thiêng W. Ví dụ, nếu W là ‘baobaab’, thì tên của thành luỹ có thể là ‘oba’, còn ‘bab’ không thể dùng để đặt tên. Dĩ nhiên không được đặt tên trùng nhau.

Tù trưởng muốn biết là có thể xây dựng được tối đa bao nhiêu thành luỹ dựa vào số tên có thể đặt.

Dữ liệu: Vào từ file văn bản BASTION.INP gồm một dòng chứa tên thiêng W, trong đó chỉ có các chữ cái la tinh thường và có độ dài không quá 1000.

Kết quả: Đưa ra file văn bản BASTION.OUT một số nguyên - số lượng tên khác nhau.

Ví dụ:

BASTION.INP		BASTION.OUT
baobaab		23



Thuật toán:

https://youtu.be/cptE_UKTcR8

11



- Đếm số lượng xâu con liên tiếp khác nhau của xâu W cho trước.
- Duyệt (i, 1, length(w))

Duyệt(j, i, length(w))

Begin

```
S:= copy(w,i, j - i + 1);  
Inc(k);  
C[k] := s; //array[1..100000] of string;
```

End;

- Sắp xếp mảng C có K phần tử thành dãy không giảm.
- Bài toán: Đếm số lượng phần tử khác nhau của mảng C đã được sắp xếp.
- Res = 0;
- For(i,1, k)
 - o If c[i] <> c[i+1] then inc(res);
- Đưa res ra.



Free Pascal IDE

```
File Edit Search Run Compile Debug Tools Options Window Help
[ ]
var
  j,i,k,res:longint;
  c:array[1..1000000] of string;
  s,w:string;
procedure qsort(L,H:longint);
var
  i,j:longint;
  tam, chot:string;
begin
  i:=L;
  j:=H;
  chot:=c[(i+j) div 2];
  while i<=j do
  begin
    while c[i]<chot do inc(i);
    while c[j]>chot do dec(j);
    if i<=j then
    begin
      tam:=c[i];
      c[i]:=c[j];
      c[j]:=tam;
      inc(i);
      dec(j);
    end;
    end;
    if L<j then qsort(L,j);
    if i<H then qsort(i,H);
  end;
BEGIN
  assign(input, 'BASTION.INP');reset(input);
  assign(output, 'BASTION.OUT');rewrite(output);
  readln(w);
  k:=0;
  for i:=1 to length(w) do
    for j:=i to length(w) do
    begin
      s:=copy(w,i,j-i+1);
      inc(k);
      c[k]:=s;
    end;
  qsort(1,k);
  res:=0;
  for i:=1 to k do
    if c[i] <> c[i+1] then inc(res);
  write(res);
  close(input);close(output);
END.
```

40:52

5) TỔNG 4 SỐ

Cho số nguyên dương x ($1 \leq x \leq 1500$). Hãy xác định số cách phân tích x thành tổng 4 số nguyên: $x = a + b + c + d$, trong đó $1 \leq a \leq b \leq c \leq d$.

Ví dụ, với $x = 6$ ta có 2 cách phân tích:

$$6 = 1 + 1 + 1 + 3,$$

$$6 = 1 + 1 + 2 + 2.$$



Dữ liệu: Vào từ file văn bản SUM.INP gồm nhiều tests, mỗi test cho trên một dòng chứa một số nguyên x .

Kết quả: Đưa ra file văn bản SUM.OUT, kết quả mỗi test đưa ra trên một dòng dưới dạng số nguyên.

Ví dụ:

SUM.INP	SUM.OUT
3	0
5	1
6	2

Thuật toán: <https://youtu.be/4HkbxTsoBGU>

- Giá trị: $1 \leq a \leq b \leq c \leq d \leq N$

- For (a,1, x)

For(b,a,x)

For(c,b,x)

For(d,c,x)

Nếu $x = a + b + c + d$ thì tăng đếm lên 1 đơn vị.

* Đưa đếm ra.

```

Free Pascal IDE

File Edit Search Run Compile Debug Tools Options
[ ] ->
var
  x, a, b, c, d, dem: longint;
begin
  assign(input,'day1_t4s.inp');reset(input);
  assign(output,'day1_t4s.out');rewrite(output);
  readln(x);
  dem := 0;
  for a := 1 to x do
    for b := a to x do
      for c := b to x do
        for d := c to x do
          if a + b + c + d = x then inc(dem);
  writeln(dem);
  close(input);close(output);
end.

```

6) TỔNG 4 SỐ NGUYÊN TỐ

Cho số nguyên dương x ($1 \leq x \leq 1500$). Hãy xác định số cách phân tích x thành tổng 4 số nguyên tố: $x = a + b + c + d$, trong đó $1 \leq a \leq b \leq c \leq d$.

Ví dụ, với $x = 8$ ta có 1 cách phân tích:

$8 = 2 + 2 + 2 + 2$;

Dữ liệu: Vào từ file văn bản SUMPRIME.INP gồm nhiều tests, mỗi test cho trên một dòng chứa một số nguyên x .

Kết quả: Đưa ra file văn bản SUMPRIME.OUT, kết quả mỗi test đưa ra trên một dòng dưới dạng số nguyên.

Ví dụ:

SUMPRIME.INP	SUMPRIME.OUT
--------------	--------------



8		1
9		1

Thuật toán:

- Tô chức hàm kiểm tra tính nguyên tố.
- Duyệt tất cả các khả năng a, b, c, d.

For (a, 1, x)

For(b, a, x)

For (c, b, x)

For (d, c, x)

Nếu $a + b + c + d = x$ và $\text{nguyento}(a)$ và $\text{nguyento}(b)$ và $\text{nguyento}(c)$ và $\text{nguyento}(d)$ thì
tăng res.

Đưa res ra.

Độ phức tạp: $O(x^4\sqrt{x})$

Cải tiến: Chuẩn bị mảng F[i] lưu trạng thái các số nguyên tố trong phạm vi từ [1, x] bằng giải thuật sàng nguyên tố Eratosthenes.

7) TÌM UỐC GCD.*

Nhập số nguyên dương N. Liệt kê các ước nguyên tố của N.

Ví dụ: N = 10; Các ước nguyên tố 2, 5.

Thuật toán: <https://youtu.be/4HkbxTsoBGU>

- $N = 26; 1, 2, 13, 26$. Có số: 2, 13.
 $= 2 \cdot 13$
- Phân tích N thành tích các thừa nguyên tố lưu vào mảng C có K phần tử.
 Procedure phantich(u: longint);
 Var i: longint;
 Begin
 i:= 2;
 While $i^2 \leq u$ do
 If $u \bmod i = 0$ then
 Begin
 Inc(k);
 C[k] := i;
 U := u div i;
 End
 Else inc(i);
 If $u > 1$ then
 Begin
 Inc(k);
 C[k] := u;
 End;
 End;
 Begin
 Readln(n);
 K:=0;



Phantich(n);
For i:=1 to k do write(c[i],#32);
End.

```
Free Pascal
File Edit Search Run Compile Debug Tools Options W:
[ ] E:\Free Pascal\bin\ttran\2022
const
  fi = 'bai2.inp';
  fo = 'bai2.out';
var n,k,i : longint;
  c : array[1..100000] of longint;
  procedure phantich(u : longint);
  var i : longint;
begin
  i := 2;
  while i*i <= u do
    if u mod i = 0 then
      begin
        k := k + 1;
        c[k] := i;
        u := u div i;
      end
    else i := i + 1;
    if u > 1 then
      begin
        k := k + 1;
        c[k] := u;
      end;
  end;
begin
  assign(input,fi); reset(input);
  assign(output,fo); rewrite(output);
  readln(n);
  k := 0;
  phantich(n);
  for i := 1 to k do write(c[i],#32);
  close(input); close(output);
end.
```

8) SỐ NGHỊCH THẾ

Xét dãy số nguyên $A = (a_1, a_2, \dots, a_n)$ ($1 \leq n \leq 100\ 000$). Các số trong dãy A khác nhau từng đôi một và nhận giá trị trong phạm vi từ 1 đến n . Như vậy dãy A là một hoán vị các số từ 1 đến n . Cặp số (a_i, a_j) trong dãy A được gọi là một nghịch thế, nếu $i < j$ và $a_i > a_j$.

Yêu cầu: Cho n và hoán vị A . Hãy xác định số nghịch thế.

Dữ liệu: Vào từ file văn bản INVERS.INP:

- Dòng đầu tiên chứa số nguyên n ,
- Dòng thứ 2 chứa n số nguyên xác định hoán vị A .

Kết quả: Đưa ra file văn bản INVERS.OUT một số nguyên – **số lượng** nghịch thế.

Ví dụ:

INVERS.INP	INVERS.OUT
5	5
2 4 3 5 1	

Thuật toán: <https://youtu.be/1t0vkycXqtI>



```
Res:=0;
For (i,1, N - 1)
    For (j, i+1, N)
        Nếu a[i] > a[j] thì inc(res);
```

Đưa Res ra.

```
Free Pascal IDE
File Edit Search Run Compile Debug Tools Options Windows
[ ] var
n,i,j,res:longint;
a:array[-1..100001] of longint;

BEGIN
    assign(input,'bai3buoi6.INP');reset(input);
    assign(output,'bai3buoi6.OUT');rewrite(output);
    readln(n);
    for i:=1 to n do read(a[i]);
    res:=0;
    for i:=1 to n-1 do
        for j:=i+1 to n do
            if a[i]>a[j] then inc(res);
    write(res);
    close(input);close(output);
END.
```

9) SỐ ĐẸP

Một số nguyên dương được gọi là số đẹp nếu **tổng các chữ số** của nó (trong hệ thập phân) chia hết cho số chữ số. Các số được xét không chứa số 0 không có nghĩa. Ví dụ, 15 là một số đẹp vì 1+5 **chia hết cho 2**.

Các số đẹp được đánh số từ 1 trở đi theo thứ tự tăng dần của giá trị.

Yêu cầu: Cho số nguyên dương n ($1 \leq n \leq 100\ 000$). Hãy tìm số đẹp thứ n .

Dữ liệu: Vào từ file văn bản BEAUTY.INP gồm nhiều tests, mỗi test ghi trên một dòng chứa một số nguyên n .

Kết quả: Đưa ra file văn bản BEAUTY.OUT, kết quả mỗi test đưa ra trên một dòng.

Ví dụ:

BEAUTY.INP	BEAUTY.OUT
1	1
15	20

Thuật toán: https://youtu.be/ZUfBsU_Y5JU

- Tổ chức hàm kiemtra(u) – có là số đẹp.
- Lần lượt thử các số p. Nếu p là số đẹp thì lưu p vào mảng a.
- Tạo đủ N phần tử của mảng a.
- Đưa a[n] ra.

Function tongcs(u: longint): longint;



Var s: longint = 0;

Begin

While u >0 do

Begin

S:= S + U mod 10;

U := u div 10;

End;

Exit(s);

End;

Function demcs(u: longint): longint;

Var s: longint = 0;

Begin

While u >0 do

Begin

S:= S + 1;

U := u div 10;

End;

Exit(s);

End;

Function kiemtra(u: longint): boolean;

Begin

If tongcs(u) mod demcs(u) = 0 then exit(true) else exit(false);

End;

function timson(u: longint): longint;

Var

Begin

A[0]:= 0;

i := 1;

while i <= u do

begin

p:= a[i-1] + 1;

while kiemtra(p) = false do inc(p);

a[i] := p;

inc(i);

end;

exit(a[u]);

End;

Chương trình chính:

Nhập N

Write(timson(n));



Free Pascal IDE

File Edit Search Run Compile Debug Tools Options Window Help

```
Day6_bai4.pas
var k,n,i,j,res:longint;
    a:array[-7..100000] of longint;
const
    fi='Day6_bai4.inp';
    fo='Day6_bai4.out';
function demcs(u:longint):longint;
var s:longint = 0;
begin
    while u>0 do
        begin
            inc(s);
            u:= u div 10;
        end;
    exit(s);
end;
function tongcs(u:longint):longint;
var s:longint = 0;
begin
    while u>0 do
        begin
            s:= s + u mod 10;
            u:= u div 10;
        end;
    exit(s);
end;
function kiemtra(u:longint):boolean;
begin
    if tongcs(u) mod demcs(u) = 0 then exit(true) else exit(false);
end;
function timson(u:longint):longint;
var p:longint;
begin
    a[0]:=0;
    i:=1;
    while i<=u do
        begin
            p:=a[i-1] + 1;
            while kiemtra(p) = false do inc(p);
            a[i]:=p;
            inc(i);
        end;
    exit(a[u]);
end;
begin
    assign(input,fi); reset(input);
    assign(output,fo); rewrite(output);
    readln(n);
    write(timson(n));
    close(input);
    close(output);
end.
```

52:17

F1 Help F2 Save F3 Open Alt+F9 Compile F9 Make Alt+F10 Local menu



Free Pascal IDE

```
[ ] File Edit Search Run Compile Debug Tools
```

```
var
  n: longint;
  a: array[-7..10007] of longint;
function tcs(u: longint): longint;
var tam: longint;
begin
  tam := 0;
  while u > 0 do
    begin
      tam := tam + (u mod 10);
      u := u div 10;
    end;
  exit(tam);
end;
function scc(u: longint): longint;
var tam: longint;
begin
  tam := 0;
  while u > 0 do
    begin
      inc(tam);
      u := u div 10;
    end;
  exit(tam);
end;
```

Free Pascal IDE

```
[ ] File Edit Search Run Compile Debug Tools
```

```
function kt(u: longint): boolean;
begin
  if tcs(u) mod scc(u) = 0 then exit(true)
  else exit(false);
end;
function ts(u: longint): longint;
var i, p: longint;
begin
  a[0] := 0;
  i := 1;
  while i <= u do
    begin
      p := a[i - 1] + 1;
      while kt(p) = false do inc(p);
      a[i] := p;
      inc(i);
    end;
  exit(a[u]);
end;
begin
  assign(input, 'day1_sdc2.inp');reset(input);
  assign(output, 'day1_sdc2.out');rewrite(output);
  readln(n);
  writeln(ts(n));
  close(input);close(output);
end.
```

F1 Help F2 Save F3 Open Alt+F9 Compile F9 M
= 47:31



10) SỐ ĐẸP 2

Một số được gọi là đẹp nếu **tổng bình phương các chữ số** của nó (trong dạng biểu diễn thập phân) là **một số nguyên tố**. Ví dụ, 12 là một số đẹp vì $1^2 + 2^2 = 5$ – số nguyên tố. Các số đẹp được đánh số theo thứ tự tăng dần của giá trị, bắt đầu từ 1 trở đi.

Yêu cầu: Cho số nguyên n ($1 \leq n \leq 10\,000$). Hãy tìm số đẹp thứ n .

Dữ liệu: Vào từ file văn bản BEAUTY2.INP, gồm nhiều tests, mỗi test cho trên một dòng chứa một số nguyên n .

Kết quả: Đưa ra file văn bản BEAUTY2.OUT, kết quả mỗi test đưa ra trên một dòng.

Ví dụ:

BEAUTY2.INP	BEAUTY2.OUT
1	11
2	12
6	23

Thuật toán: <https://youtu.be/KTrVmNESzoI>

- Tổ chức hàm kiemtra(u) – có là số đẹp.
- Lần lượt thử các số p. Nếu p là số đẹp thì lưu p vào mảng a.
- Tạo đủ N phần tử của mảng a.
- Đưa a[n] ra.

```
Function binhphuongcs(u: longint): longint;
Var s: longint = 0;
Begin
  While u > 0 do
    Begin
      S := S + sqr(U mod 10);
      U := u div 10;
    End;
    Exit(s);
End;
```

```
Function nguyento(u: longint): boolean;
Var j: longint;
Begin
  If u < 2 then exit(false);
  For j:=2 to trunc(sqrt(u)) do
    If u mod j = 0 then exit(false);
  Exit(true);
End.
```

```
function kiemtra(u: longint): boolean;
begin
  if nguyento(binhphuongcs(u)) = true then exit(true) else exit(false);
end;

function timson(u: longint): longint;
Var
```



Begin

```
A[0]:= 0;
i := 1;
while i <= u do
begin
    p:= a[i-1] + 1;
    while kiemtra(p) = false do inc(p);
    a[i] := p;
    inc(i);
end;
exit(a[u]);
```

End;

Chương trình chính:

Nhập N

Write(timson(n));

```
Free Pascal IDE
File Edit Search Run Compile D
[ ]—
var
  n: longint;
  a: array[-7..10007] of longint;
function snt(u: longint): boolean;
var j: longint;
begin
  if u <= 1 then exit(false);
  for j := 2 to trunc(sqrt(u)) do
    if u mod j = 0 then exit(false);
  exit(true);
end;
function tcs(u: longint): longint;
var tam: longint;
begin
  tam := 0;
  while u > 0 do
  begin
    tam := tam + (u mod 10);
    u := u div 10;
  end;
  exit(tam);
end;
```



The screenshot shows a Free Pascal IDE window with the following code:

```
Free Pascal IDE
File Edit Search Run Compile Debug Tool
[ ]
function bpcs(u: longint): longint;
var tam: longint;
begin
    tam := 0;
    while u > 0 do
        begin
            tam := tam + sqr(u mod 10);
            u := u div 10;
        end;
    exit(tam);
end;
function kt(u: longint): boolean;
begin
    if snt(bpcs(u)) = true then exit(true)
    else exit(false);
end;
function ts(u: longint): longint;
var i, p: longint;
begin
    a[0] := 0;
    i := 1;
    while i <= u do
        begin
            p := a[i - 1] + 1;
            while kt(p) = false do inc(p);
            a[i] := p;
            inc(i);
        end;
    exit(a[u]);
end;
begin
    assign(input,'day1_sdc2.inp');reset(input);
    assign(output,'day1_sdc2.out');rewrite(output);
    readln(n);
    writeln(ts(n));
    close(input);close(output);
end.
```

At the bottom of the IDE window, the status bar shows "59:37" and the menu bar includes F1 Help, F2 Save, F3 Open, Alt+F9 Compile, F9 Make, and Alt.

11) TÍCH CÁC CHỮ SỐ

Cho hai số nguyên l và r thỏa mãn điều kiện $1 \leq l \leq r \leq 10^9$, $r - l \leq 10^5$.

Yêu cầu: Hãy xác định **số lượng** các số nguyên trong khoảng $[l, r]$ **chia hết cho tích các chữ số** của mình (trong dạng biểu diễn thập phân).

Dữ liệu: Vào từ file văn bản PRODUCT.INP gồm một dòng chứa hai số nguyên l và r .

Kết quả: Đưa ra file văn bản PRODUCT.OUT số lượng số tìm được.

Ví dụ:

PRODUCT.INP		PRODUCT.OUT
1 12		11

Thuật toán:

<https://youtu.be/u6AIXaYTSt4>



- Tổ chức hàm tính tích chữ số của u.

- For (u, L, R) nếu $tichcs(u) > 0$ thì
 - Nếu $u \bmod tichcs(u) = 0$ thì tăng đếm lên.
- Đưa đếm ra.

```
Function tichcs(u: longint): longint;
```

```
Var s: longint = 1;
```

```
Begin
```

```
    While u > 0 do
```

```
        Begin
```

```
            S:= S * (U mod 10);
```

```
            U := u div 10;
```

```
        End;
```

```
        Exit(s);
```

```
    End;
```

```
Free Pascal
```

```
File Edit Search Run Compile Debug Tools Options W
[ ] E:\Free Pascal\bin\ttran\202
const
    fi = 'bai6.inp';
    fo = 'bai6.out';
var l,r : longint;
    a : array[-7..100007] of longint;
    function tichcs(u : longint): longint;
    var s : longint;
    begin
        s := 1;
        while u > 0 do
        begin
            s := s * (u mod 10);
            u := u div 10;
        end;
        exit(s);
    end;
    procedure kiemtra(l,r : longint);
    var res,u : longint;
    begin
        res := 0;
        for u := l to r do
        if tichcs(u) <> 0 then
            if u mod tichcs(u) = 0 then
                res := res + 1;
        writeln(res);
    end;
begin
    assign(input,fi); reset(input);
    assign(output,fo); rewrite(output);
    readln(l,r);
    kiemtra(l,r);
    close(input); close(output);
end.
```



12) PHƯƠNG TRÌNH

Cho phương trình

24



$$\frac{ax+b}{cx+d} = v$$

Trong đó x là ẩn số của phương trình, còn a, b, c, d, v là các số nguyên, mỗi số có giá trị tuyệt đối không vượt quá 1 000.

Yêu cầu: Tìm và đưa ra nghiệm phương trình dưới dạng $X = p/q$, trong đó p, q là các số nguyên và nguyên tố cùng nhau. Nếu phương trình vô nghiệm thì đưa ra thông báo **NONE**, trong trường hợp vô định – đưa ra thông báo **MULTIPLE**.

Dữ liệu: Vào từ file văn bản EQUATION.INP gồm nhiều tests, mỗi test cho trên một dòng chứa 5 số nguyên a, b, c, d và v .

Kết quả: Đưa ra file văn bản EQUATION.OUT, mỗi kết quả đưa ra trên một dòng.

Ví dụ:

EQUATION.INP		EQUATION.OUT
1 2 3 4 5		X = -9/7
1 1 1 1		MULTIPLE

Thuật toán: <https://youtu.be/v-pK5j938ss>

- Biến đổi toán học đưa về phương trình bậc nhất một ẩn:
 $(a-v.c)x + (b - v.d) = 0$; TXĐ: $x \neq -d/c$.
- Ta đặt $p = a - v.c; q = b - v.d$; phương trình $P.x + q = 0$.
- Cài hàm tìm ucln(u, v).
- Giải và biện luận phương trình bậc nhất một ẩn
 - o Nếu $p = 0$ và $q = 0$ thì write('MULTIPLE');
 - o Nếu $p = 0$ và $q \neq 0$ thì write('NONE');
 - o Nếu $p \neq 0$ thì
 - Nếu $-q/p = -d/c$ thì write('NONE')
 - Ngược lại
 - Begin
 - Nếu $p*q > 0$ thì dau := 1 else dau := -1;
 - $P := \text{abs}(p); q := \text{abs}(q);$
 - $T := \text{ucln}(p, q);$
 - $P := P \text{ div } T;$
 - $Q := q \text{ div } T;$
 - Write(dau*q, #32, p);
 - End;

13) DÃY SỐ

Xét dãy số nguyên $a_1, a_2, \dots, a_n, \dots$, trong đó $a_1 = 1$, a_n được xác định như sau: **đảo ngược thứ tự viết các chữ số của a_{n-1}** (trong hệ cơ số 10) và cộng thêm 2 vào số nhận được.

Phần đầu của dãy số này có giá trị như sau:

Chỉ số	1	2	3	4	5	6	7	8	9	10	11	12	...
Dãy a	1	3	5	7	9	11	13	33	35	55	57	77	...



Yêu cầu: Cho số nguyên dương n . Hãy xác định a_n ($1 \leq n \leq 10^{12}$).

Dữ liệu: Vào từ file văn bản SEQUENCE.INP gồm nhiều tests, mỗi test cho trên một dòng chứa một số nguyên n .

Kết quả: Đưa ra file văn bản SEQUENCE.OUT, kết quả mỗi test đưa ra trên một dòng.

Ví dụ:

SEQUENCE.INP	SEQUENCE.OUT
1	1
12	77

Thuật toán: <https://youtu.be/HROFOzOms0E>

Tổ chức hàm tìm số đảo của u .

```
Function sodao(u: longint): longint;
```

```
Var S: longint;
```

```
Begin
```

```
    S:= 0;
```

```
    While u > 0 do
```

```
        Begin
```

```
            S:= S*10 + U mod 10;
```

```
            U := u div 10;
```

```
        End;
```

```
        Exit(s);
```

```
    End;
```

- $A[1] := 1$;

- Duyệt($i, 2, N$) $a[i] = \text{sodao}(a[i-1]) + 2$;

- Đưa $a[n]$ ra.

```
Free Pascal IDE
File Edit Search Run Compile Debug Tools Options Window
=[ ] sodao.p
var n,i:longint;
    a:array[1..100000] of longint;
function sodao(u:longint):longint;
var s:longint;
begin
    s:=0;
    while u>0 do
        begin
            s:=s*10+u mod 10;
            u:=u div 10;
        end;
    exit(s);
end;
begin
    assign(input,'saodao.inp'); reset(input);
    assign(output,'sodao.out'); rewrite(output);
    readln(n);
    a[1]:=1;
    for i:=2 to n do
        a[i]:=sodao(a[i-1])+2;
    writeln(a[n]);
    close(input);close(output);
end.
```



14) MÔ ĐUN M

Cho n số nguyên a_1, a_2, \dots, a_n ($|a_i| < 10^9$, $0 \leq n \leq 100\,000$). Hãy xác định dãy con nhiều phần tử nhất từ dãy đã cho, sao cho không có hai phần tử nào của dãy con có tổng chia hết cho m ($2 \leq m \leq 100\,000$).

Dữ liệu: Vào từ file văn bản MODM.INP:

- Dòng thứ nhất chứa 2 số nguyên n và m ,
- Dòng thứ 2 chứa n số nguyên a_1, a_2, \dots, a_n .

Kết quả: Đưa ra file văn bản MODM.OUT:

- Dòng thứ nhất chứa số nguyên k – số phần tử của dãy con tìm được,
- Dòng thứ 2 chứa k số nguyên – chỉ số trong dãy ban đầu của các phần tử thuộc dãy con.

Nếu có nhiều kết quả thì đưa ra một trong số chúng.

Ví dụ:

MODM.INP		MODM.OUT
3 2 1 100 10		2 1 2

Thuật toán:

<https://youtu.be/Z9-RS9I7U0E>

- Sinh dãy nhị phân độ dài N vào mảng b.
- Với mỗi cấu hình, xuli()

K:=0;

Duyệt(i,1,N) nếu $b[i] = 1$ thì

Begin

 Inc(k); c[k] := a[i];

End;

For i:=1 to k-1 do

 For j:=i+1 to k do

 If (c[i] + c[j]) mod m = 0 then exit();

If k > res then

Begin

 Res:= k;

 For i:=1 to n do d[i] := b[i];

End;

- **Chương trình chính:**

Nhập N và dãy a1, a2, ..., aN.

Res := 0;

Thu(1);

Đưa res ra.

For(i,1,n)

 If d[i] = 1 then write(i,#32);



Free Pascal IDE

```
File Edit Search Run Compile Debug Tools Options View
=[■]= Day7_bai3.pas
var x,res,n,i,k,m:longint;
    a,b,c,d:array[1..100000] of longint;
const
    fi='Day7_bai3.inp';
    fo='Day7_bai3.out';
procedure xuly();
var i,j:longint;
begin
    k:=0;
    for i:=1 to n do
        if b[i] = 1 then
            begin
                inc(k);
                c[k]:=a[i];
            end;
    for i:=1 to k-1 do
        for j:=i+1 to k do
            if (c[i] + c[j]) mod m = 0 then exit();
    if k> res then
    begin
        res:=k;
        for i:=1 to n do d[i]:=b[i];
    end;
end;
procedure thu(i:longint);
var j:longint;
begin
    if i>=n then exit();
    for j:=i+1 to n do
        if (d[j] - d[i]) mod m = 0 then
            begin
                d[i]:=d[j];
                thu(j);
            end;
end;
begin
    readln(n);
    for i:=1 to n do
        read(a[i]);
    for i:=1 to n do
        read(b[i]);
    for i:=1 to n do
        read(d[i]);
    res:=0;
    xuly();
    writeln(res);
    thu(1);
    writeln(d[1]);
end.
```



Free Pascal IDE

File Edit Search Run Compile Debug Tools Options W: Day7_bai3.pas

```
for i:=1 to k-1 do
    for j:=i+1 to k do
        if (c[i] + c[j]) mod m = 0 then exit();
if k > res then
begin
    res:=k;
    for i:=1 to n do d[i]:=b[i];
end;
end;
procedure thu(i:longint);
var j:longint;
begin
    for j:=0 to 1 do
        begin
            b[i]:=j;
            if i=n then xuly()
            else thu(i+1);
        end;
end;
begin
    assign(input,fi); reset(input);
    assign(output,fo); rewrite(output);
    readln(n,m);
    for i:=1 to n do read(a[i]);
    res:=0;
    thu(1);
    writeln(res);
    for i:=1 to n do
        if d[i] = 1 then write(i,#32);
    close(input); close(output);
end.
```

47:57 F1 Help F2 Save F3 Open Alt+F9 Compile F9 Make Alt+F10

15) ĐẦU TƯ CHỨNG KHOÁN

Harry làm việc ở một công ty tư vấn đầu tư chứng khoán. Nhiệm vụ của Harry là phân tích sự giao động chỉ số chứng khoán hàng ngày của sàn giao dịch, từ đó có thể các thông tin hữu ích để tư vấn cho các nhà đầu tư.

Nhiệm vụ của Harry trong đề án đang thực hiện là như sau: Cho dãy chỉ số chứng khoán của n ngày liên tiếp. **Cần phải tìm ra dãy con dài nhất sao cho chênh lệch hai chỉ số liên tiếp không ít hơn k .** Ví dụ, với dãy chỉ số chứng khoán 1014, 1024, 1034, 1045, 1030, 998 và $k = 15$ thì dãy con 1014, 1034, 998 là chấp nhận được, nhưng dãy 1014, 1045, 1030, 998 là dãy con dài nhất cần tìm.

Yêu cầu: Cho n, k và dãy các chỉ số chứng khoán ($1 \leq n \leq 100\,000, 1 \leq k \leq 10^9$). Các chỉ số chứng khoán là những số nguyên dương, có giá trị không vượt quá 10^9 . Hãy chỉ ra dãy con dài nhất thỏa mãn các điều kiện đã nêu.

Dữ liệu: Vào từ file văn bản FINANCIAL.INP

- Dòng đầu tiên chứa hai số nguyên n và k ,
- Dòng thứ 2 chứa n số nguyên – các chỉ số chứng khoán.

Kết quả: Đưa ra file văn bản FINANCIAL.OUT:



- Dòng đầu tiên đưa ra độ dài của dãy con tìm được,
 - Dòng thứ hai – các chỉ số thuộc dãy con theo trình tự xuất hiện.
- Nếu có nhiều dãy con cùng độ dài thì đưa ra dãy tùy ý.

Ví dụ:

FINANCIAL.INP		FINANCIAL.OUT
6 15 1014 1024 1034 1045 1030 998		4 1014 1045 1030 998

Thuật toán:

- <https://youtu.be/zxKcJpFFLmQ>
- Sinh dãy nhị phân độ dài N vào mảng b.
- Với mỗi cấu hình, xuli()

K:=0;

Duyệt(i,1,N) nếu b[i] = 1 thì

Begin

 Inc(k); c[k] := a[i];

End;

For i:=1 to k-1 do

 If **abs(c[i] - c[i+1]) < m** then exit();

If k > res then

Begin

 Res:= k;

 For i:=1 to n do d[i] := b[i];

End;

- **Chương trình chính:**

Nhập N, M và dãy a1, a2, ..., aN.

Res := 0;

Thu(1);

Đưa res ra.

For(i,1,n)

 If d[i] = 1 then write(a[i],#32);



```
1 #include <bits/stdc++.h>
2 #define nmax 100007
3 using namespace std;
4 int a[nmax], b[nmax], c[nmax], d[nmax];
5 int n, m, k, res=0;
6
7 void xl()
8 {
9     k=0;
10    for(int i=1;i<=n;i++)
11        if(b[i]==1)
12        {
13            k++;
14            c[k]=a[i];
15        }
16    for(int i=1;i<=k-1;i++)
17        if(abs(c[i]-c[i+1])<m)
18            return;
19    if(k>res)
20    {
21        res =k;
22        for(int i=1;i<=n;i++)
23            d[i]=b[i];
24    }
25 }
26 void thu(int i)
27 {
28     for(int j=0;j<=1;j++)
29     {
30         b[i]=j;
31         if(i==n)
32             xl();
33         else thu(i+1);
34     }
35 }
36 int main()
37 {
38     freopen("FINANCIAL.inp","r",stdin);
39     freopen("FINANCIAL.out","w",stdout);
40     cin>>n>>m;
41     for(int i=1;i<=n;i++)
42         cin>>a[i];
43     thu(1);
44     cout<<res<<endl;
45     for(int i=1;i<=n;i++)
46         if(d[i]==1)
47             cout<<a[i]<<" ";
48     return 0;
49 }
50 }
```



Free Pascal IDE

```

File Edit Search Run Compile Debug Tools Options
procedure thu(u: longint);
var
  j: longint;
begin
  for j:= 1 to n do
  begin
    b[u]:= j;
    if u = n then xuli()
    else thu(u+1);
  end;
end;
begin
  assign(input,'financial.inp'); reset(input);
  assign(output,'financial.out'); rewrite(output);
  readln(n,m);
  for i:= 1 to n do read(a[i]);
  res:= 0;
  thu(1);
  writeln(res);
  for i:= 1 to n do
    if d[i] = 1 then write(a[i],#32);
  close(input);
  close(output);
end.

```

16) CHUỖI NHỊ PHÂN

Xét xâu nhị phân, tức là xâu chỉ chứa các ký tự trong tập $\{0, 1\}$. Gọi k là số lượng xâu nhị phân độ dài n ($1 \leq n \leq 10^4$) chứa xâu S (độ dài không quá 100) như một xâu con (các ký tự liên tiếp) đúng một lần.

Yêu cầu: Hãy tính phần dư của kết quả chia k cho $10^9 + 7$.

Dữ liệu: Vào từ file văn bản BINARY.INP:

- Dòng thứ nhất chứa số nguyên n ,
- Dòng thứ hai chứa xâu S .

Kết quả: Đưa ra file văn bản BINARY.OUT một số nguyên – kết quả tìm được.

Ví dụ: // p32 Ind9 _0103/2008A

BINARY.INP	BINARY.OUT
4	10
01	

Thuật toán:

<https://youtu.be/rrsSKS1Msfs>

Đếm số xâu nhị phân độ dài N chứa xâu S đúng 1 lần.

- Cho xâu S_0 và xâu S_1 . Viết hàm kiểm tra xâu S_1 có xuất hiện đúng 1 lần trong xâu S hay không?
- Nếu có trả về True ngược lại trả về False.
- Lần lượt sinh dãy nhị phân độ dài N vào xâu S_0 .
- const L = trunc(1e9+7);
- **S0 : array[1..30 of char;**



```
Function kiemtra(u,v: string):boolean; // kiem tra xau u xuat hien trong xau v đúng 1 lan.  
Var dem: longint;  
Begin  
    Dem:=0;  
    While (pos(u,v))> 0 do  
        Begin  
            inc(dem);  
            If dem > 1 then exit(false);  
            Delete(v,1, pos(u,v) + length(u) - 1 );  
        end;  
        exit(dem = 1);  
    End;  
    Procedure xuli();  
    Begin  
        If (kiemtra(s,s0)) then inc(res);  
        Res := res mod L;  
    End;  
    Procedure thu(i: longint);  
    Var j: char;  
    Begin  
        For j:='0' to '1' do  
            Begin  
                S0[i] := j;  
                if i = n then xuli()  
                else thu(i+1);  
            End;  
    End;
```



Free Pascal IDE

File Edit Search Run Compile Debug Tools Options

```
[ ]
```

```
const L = trunc(1e9 + 7);
var
  n, res, ii: longint;
  s: string;
  s0: array[1..30] of char;
function kt(u, v: string): boolean;
var dem: longint;
begin
  dem := 0;
  while (pos(u, v)) > 0 do
    begin
      inc(dem);
      if dem > 1 then exit(false);
      delete(v, 1, pos(u, v) + length(u) - 1);
    end;
  exit(dem = 1);
end;
procedure xuli();
var p: longint;
begin
  if (kt(s, s0)) then inc(res);
  res := res mod l;
end;
procedure thu(i: longint);
var j: char;
begin
  for j := '0' to '1' do
    begin
      s0[i] := j;
      if i = n then xuli
      else thu(i + 1);
    end;
end;
begin
  assign(input, 'binary.inp'); reset(input);
  assign(output, 'binary.out'); rewrite(output);
  readln(n);

  readln(s);
  res := 0;
  thu(1);
  write(res);
  close(input); close(output);
end.
```

27:52

F1 Help F2 Save F3 Open Alt+F9 Compile F9 Make

17) LŨY THÙA 2

Xét tập T chỉ chứa các số nguyên có tích các chữ số (trong hệ thập phân) là **một số lũy thừa của 2**. Các số này được sắp xếp theo thứ tự tăng dần của giá trị và đánh số bắt đầu từ 1 trở đi.

Yêu cầu: Cho số nguyên k ($1 \leq k \leq 10^{18}$). Hãy tìm số thứ k trong tập T .



Dữ liệu: Vào từ file văn bản POWER2.INP, gồm nhiều tests, mỗi test cho trên một dòng chứa một số nguyên k .

Kết quả: Đưa ra file văn bản POWER2.OUT, kết quả mỗi test đưa ra trên một dòng.

Ví dụ:

POWER2.INP	POWER2.OUT
1	1
10	22

Thuật toán: <https://youtu.be/a-si5iVDAjw>

- Tìm phần tử thứ K của mảng $a[i]$. Mảng $a[i]$ được xây dựng theo yêu cầu.

```
Function tichcs(u: longint): longint;
```

```
Var s: longint = 1;
```

```
Begin
```

```
    While u > 0 do
```

```
        Begin
```

```
            S:= s*(U mod 10);
```

```
            U:= u div 10;
```

```
        End;
```

```
        Exit(s);
```

```
    End;
```

```
Function kiemtra(u: longint): boolean; //Kiểm tra U có là lũy thừa của 2
```

```
Begin
```

```
    If u = 0 then exit(false);
```

```
    While u > 1 do
```

```
        If u mod 2 = 0 then u:= u div 2
```

```
        Else break;
```

```
    If u > 1 then exit(false) else exit(true);
```

```
End;
```

```
Function timson(u: longint): longint;
```

```
Var p,i: longint;
```

```
Begin
```

```
    A[0]:= 0;
```

```
    I:= 1;
```

```
    While i <= u do
```

```
        Begin
```

```
            P:= a[i-1] + 1;
```

```
            While kiemtra(tichcs(p)) = false do in(p);
```

```
            A[i] := p;
```

```
            Inc(i);
```

```
        End;
```

```
        Exit(a[u]);
```

```
    End;
```

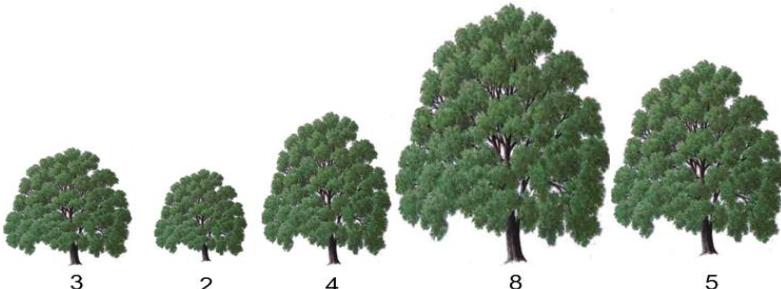
```
    Readln(n);
```

```
    Write(timson(n));
```



18) CÂY SỐI

Từ cổng vào đến tòa chính của Bộ Quốc phòng có trồng một hàng n cây sồi ($2 \leq n \leq 200$). Các cây được đánh số từ 1 đến n từ trái sang phải.



Để chuẩn bị đón Tổng tham mưu trưởng đến nhậm chức Bộ trưởng ra lệnh chặt bớt một số cây để dãy các cây còn lại thể hiện sắc nét hơn tính kỷ luật của một tổ chức quân sự. Chỉ thị nội bộ chỉ cho phép chặt một cây trong hai trường hợp:

- Cây sát ngay bên phải và cây sát ngay bên trái thực sự thấp hơn cây này,
- Cây sát ngay bên phải và cây sát ngay bên trái thực sự cao hơn cây này.

Như vậy, theo chỉ thị cây bên trái nhất và cây bên phải nhất của hàng sẽ không bị chặt.

Bộ trưởng yêu cầu lên kế hoạch chặt để trong hàng cây còn lại, mỗi cây sẽ không thấp hơn tất cả các cây bên trái nó trong hàng. Là một người yêu thiên nhiên, Bộ trưởng yêu cầu phải tìm cách chặt ít cây nhất.

Yêu cầu: Cho n và độ cao h_i của cây thứ i ($1 \leq h_i \leq 1000$, $i = 1 \dots n$). Hãy xác định xem có thể chặt để tạo ra dãy cây như mong muốn hay không, nếu có thì chỉ ra số cây và các cây cần chặt.

Dữ liệu: Vào từ file văn bản OAKS.INP:

- Dòng đầu tiên chứa số nguyên n ,
- Dòng thứ 2 chứa các số h_1, h_2, \dots, h_n .

Kết quả: Đưa ra file văn bản OAKS.OUT: Nếu không có phương án chặt thì đưa ra số -1, trong trường hợp ngược lại:

- Dòng đầu tiên đưa ra số nguyên k – số cây cần chặt,
- Mỗi dòng trong k dòng sau chứa một số nguyên xác định cây cần chặt.

Ví dụ:

OAKS.INP		OAKS.OUT
5		2
3 2 4 8 5		2
		4

Thuật toán:

- <https://youtu.be/UUyB4eNX6sQ>
- Hàm kiểm tra mảng C có K phần tử có là dãy không giảm?
- Hàm kiểm tra nút thứ i có xoá được hay không?
- Xuli()
 - Đãy các phần tử được chọn vào mảng C có K phần tử.
 - Nếu kiemtra() = true và kiểm tra các phần tử có bit b[i]=0 thì có thỏa mãn điều kiện xoá.
- Thu(i: longint);

Function kiemtra(): boolean; //Kiem tra mang C la day khong giam

Var i: longint;



```
Begin
    For i:=1 to k - 1 do
        If c[i] > c[i+1] then exit(false);
        Exit(true);
End;

Function kiemtranut(u:longint): boolean; //kiem tra a[u] co xoa duoc hay khong
Begin
    Exit (((A[u] > a[u-1]) and (a[u] > a[u +1])) or ((A[u] < a[u-1]) and (a[u] < a[u +1])));
End;

Procedure xuli();
Var i: longint;
Begin
    For i:=1 to n do
        If b[i] = 0 then
            If kiemtranut(i) = false then exit;
        K:= 0;
        For i:=1 to n do
            If b[i] = 1 then
                Begin
                    Inc(k);
                    C[k]:= a[i];
                End;
            If (kiemtra()) and (res < k) then
                Begin
                    Res := k;
                    For i:= 1 to n do d[i] := a[i];
                End;
        End;
End;

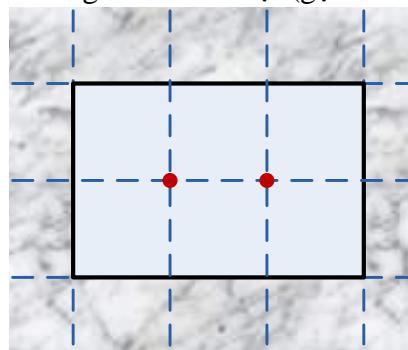
Procedure thu(i: longint);
.....
.
....
.
.....
Writeln(N-res); // số lượng phần tử bị xoá ít nhất
For(i,1,N)
    If d[i] = 0 then write(i,#32);
```



19) CHỮ NHẬT

Steva vẽ một hình chữ nhật trên giấy kẻ ô vuông, cạnh của hình chữ nhật trùng với đường chia lưới ô vuông, sau đó đếm số *điểm nút* nằm hẳn bên trong hình chữ nhật (gọi số đó là K) và số *cạnh* nằm hẳn bên trong hình chữ nhật của các ô vuông lưới (gọi số đó là L). Trong trường hợp ở hình bên Steva có $K = 2$, $L = 7$. Steva ghi lại các giá trị K , L vào sổ ghi chép. Vài hôm sau, tình cờ thấy lại các số liệu này Steva không thể nào nhớ nổi kích thước m , n của lưới ô vuông mình đã vẽ. Tuy vậy Steva tin rằng với hai giá trị đã biết thì có thể tìm được hai ẩn số m và n .

Yêu cầu: Cho hai số nguyên K và L ($0 \leq K, L \leq 1\,000$). Hãy đưa ra một bộ giá trị m , n phù hợp. Dữ liệu đảm bảo có nghiệm.



Dữ liệu: Vào từ file văn bản RECT.INP, gồm nhiều tests, mỗi test cho trên một dòng chứa 2 số nguyên K , L .

Kết quả: Đưa ra file văn bản RECT.OUT, kết quả mỗi test đưa ra trên một dòng chứa 2 số nguyên m và n .

Ví dụ:

RECT.INP
2 7
1 4

RECT.OUT
2 3
2 2

20) SỐ LUỢNG

Với mỗi số nguyên dương n , người ta có thể xác định hai đại lượng p và q , trong đó:

- p là tích các chữ số của n ,
- $q = n \times p$.

Ví dụ, với $n = 2612$, ta có $p = 2 \times 6 \times 1 \times 2 = 24$, $q = 2612 \times 24 = 62688$.

Yêu cầu: Cho hai số nguyên dương a và b ($1 \leq a \leq b \leq 10^{18}$). Hãy xác định số lượng số n có q tương ứng thuộc $[a, b]$.

Dữ liệu: Vào từ file văn bản QUANTITY.INP gồm một dòng chứa hai số nguyên a và b .

Kết quả: Đưa ra file văn bản QUANTITY.OUT số lượng tìm được.

Ví dụ: // q01Croatia 26/04/2008

QUANTITY.INP		QUANTITY.OUT
145 192		4

21) LỚP HỌC MÚA

Lớp học múa khiêu vũ dạ hội của giáo sư Padegras có n học sinh nam và nữ ghi tên. Giáo sư cho tất cả học sinh xếp thành một hàng dọc và chọn một nhóm các học sinh liên tiếp nhau cho buổi học đầu tiên với yêu cầu là số học sinh nam và nữ phải bằng nhau.

Hãy xác định, giáo sư Padegras có **bao nhiêu cách lựa chọn** khác nhau cho buổi học đầu tiên.

Dữ liệu: Vào từ file văn bản DANCE.INP:

- Dòng đầu tiên chứa số nguyên n ($1 \leq n \leq 10^6$),
- Dòng thứ 2 chứa xâu độ dài n bao gồm các ký tự từ tập $\{a, b\}$ xác định dòng xếp hàng, a là nam, b – nữ.

Kết quả: Đưa ra file văn bản DANCE.OUT một số nguyên – số cách lựa chọn.



Ví dụ:



DANCE.INP	DANCE.OUT
8	13
abbababa	

22) DÃY SỐ HAMMING

Dãy số nguyên dương tăng dần, trong đó ước nguyên tố của mỗi số không quá 5 được gọi là dãy Hamming. Như vậy, $10 = 2 \times 5$ sẽ là một số trong dãy Hamming, còn $26 = 2 \times 13$ – không thuộc dãy Hamming.

Phân đầu của dãy Hamming là $1, 2, 3, 4, 5, 6, 8, 9, 10, 12, 15, \dots$

Yêu cầu: Cho số nguyên x ($1 \leq x \leq 10^{18}$). Hãy xác định số thứ tự của x trong dãy Hamming.

Đữ liệu: Vào từ file văn bản HAMMING.INP:

- Dòng đầu tiên chứa số nguyên t – số lượng tests ($1 \leq t \leq 10^5$),
- Mỗi dòng tiếp theo chứa một số nguyên x .

Kết quả: Đưa ra file văn bản HAMMING.OUT, kết quả mỗi test đưa ra trên một dòng dưới dạng số nguyên hoặc thông báo *Not in sequence*.

Ví dụ:

HAMMING.INP	HAMMING.OUT
11	1
1	2
2	6
6	Not in sequence
7	7
8	8
9	9
10	Not in sequence
11	10
12	Not in sequence
13	Not in sequence
14	

Thuật toán:

- <https://youtu.be/bULTcFYeXzw>
- Lần lượt sinh ra các số Hamming lưu vào mảng $a[i]$.
- Nếu $a[i] = X$ thì $\text{exit}(i)$;
- Nếu $a[i] > X$ thì $\text{exit}(0)$;
- Chương trình chính: nếu $\text{timson}(x) = 0$ thì $\text{write}(\text{'Not in sequence'})$
Ngược lại thì đưa $\text{timson}(x)$ – vị trí của x trong dãy Hamming

Function kiemtra(u: longint): boolean;

Var p: longint;

Begin

P:= 2;

While $p * p \leq u$ do

If $u \bmod p = 0$ then

Begin



```
If p > 5 then exit(false);
U := u div p;
End
Else
Inc(p);
If (u > 1)
If (u > 5) exit(false);

Exit(true);
End;

Function timson(u: longint): longint; //tim stt cua u trong day Hamming
Var p,i: longint;
Begin
A[0]:= 0;
I:= 1;
While a[i-1] <= u do
Begin
P:= a[i-1] + 1;
While kiemtra(p) = false do in(p);
A[i] := p;
If a[i] = u then break;
Inc(i);
End;
If a[i] = u then exit(i) else exit(0);
End;

Readln(n);
q:= timson(n);
If q = 0 then write('Not in sequence') else write(q);
```

```
1 var n,q:longint;
2 a:array[-5..100007] of longint;
3 function check(x:longint):boolean;
4 var p:longint;
5 begin
6   p:=2;
7   while p*p <= x do
8     if x mod p = 0 then
9       begin
10      if p > 5 then exit(false);
11      x:=x div p;
12    end
13    else inc(p);
14  if x > 5 then exit(false);
15  exit(true);
16 end;
```



```
17  function findint(x:longint):longint;
18  var p,i: longint;
19  begin
20      a[0]:=0;
21      i:=1;
22      while a[i-1] <= x do
23      begin
24          p:=a[i-1] + 1;
25          while check(p) = false do inc(p);
26          a[i]:=p;
27          if a[i] = x then break;
28          inc(i);
29      end;
30      if a[i] = x then exit(i) else exit(0);
31  end;
32  begin
33      readln(n);
34      q:=findint(n);
35      if q = 0 then write('Not in sequence') else write(q);
36  end.
```



```
1 #include <bits/stdc++.h>
2 #define nmax 100007
3 using namespace std;
4 int f[nmax];
5 long s;
6 bool snt(int u)
7 {
8     if(u<2) return false;
9     if(u==2 || u==3) return true;
10    for(int j=2;j<=trunc(sqrt(u));j++)
11        if(u%j==0) return false;
12    return true;
13 }
14 void sgnt(int u)
15 {
16     for(int i=1;i<=u;i++) f[i]=1;
17     f[1]=0;
18     for(int i=2;i*i<=u;i++)
19         if(f[i]==1)
20             for(int j=i;j*j<=u;j+=i)
21                 f[i*j]=0;
22 }
23 int main()
24 {
25     int n;
26     cin>>n;
27     sgnt(n);
28     for(int i=1;i<=n;i++)
29         if(f[i]==1)
30             s+=i;
31     cout<<s<<endl;
32     if(snt(s)) cout<<"YES";
33     else cout<<"NO";
34     return 0;
35 }
36 }
```

23) TỔNG NGUYÊN TỐ

Tổng các số nguyên tố từ 2 đến 1 000 000

là 37550402023 và thực sự cũng là một số nguyên tố.

Từ blog <http://blog.sjinks.org.ua>

Vaxia sinh hoạt trong tổ ngoại khóa Toán. Mới đây, được giới thiệu về số nguyên tố, Vaxia hào hứng lên Internet tìm thêm thông tin với từ khóa tìm kiếm Prime numbers. Một trong những thông tin tìm thấy được nêu ở trên!

Vaxia tự hỏi: "Còn số tự nhiên n nào nữa để tổng các số nguyên tố trong phạm vi từ 2 tới n cũng là một số nguyên tố?" Và dĩ nhiên, để trả lời câu hỏi đó chỉ có thể dùng máy tính.

Dữ liệu: Vào từ file văn bản PRIME.INP số nguyên n ($2 \leq n \leq 10^4$).

Kết quả: Đưa ra file văn bản PRIME.OUT:

- Dòng thứ nhất: tổng tìm được,



- Dòng thứ 2: thông báo **YES** nếu tổng tìm được là số nguyên tố, trong trường hợp ngược lại – **NO**.

Ví dụ:

PRIME.INP	PRIME.OUT
3	5
	YES

Thuật toán: <https://youtu.be/2GEifQIb5OU>

Subtask1:

Duyệt (i, 2, N)

nếu nguyento(i) = true thì s := s + i;

Đưa S ra.

Nếu nguyento(s) = true thì write('YES') else write('NO');

Độ phức tạp thuật toán: $O(N\sqrt{N})$. Với $N \leq 10000$.

Subtask2:

- Sàng nguyên tố Eratosthenes (N) thu được mảng F[i] lưu trạng thái các số nguyên tố thuộc đoạn [1, N].
- Duyệt (i, 2, N)
 - nếu F[i] = true thì s := s + i;
 - Đưa S ra.
 - Nếu nguyento(S) = true thì write('YES') else write('NO');

Độ phức tạp: $O(N + \sqrt{S}) \sim O(N)$ chạy được với $N \leq 10^6$.

Function nguyento(u: longint): boolean;

,....

Procedure sangnt(u: longint);

Var i,k: longint;

Begin

For i:=1 to u do f[i] := true;

F[1] := false;

I:= 2;

While i*i <= u do

If f[i] = true then

Begin

K:=2;

While k*i<= u do

Begin

F[k*i]:= false;

K:= k+1;

End;

Inc(i);

End;

End;



```
Begin
```

```
//doc du lieu
Sangnt(n);
S:=0;
For(i, 2, N)
    If f[i] = true then s := s + i;
Writeln(s);
If nguyento(s) = true then write('YES') else write('NO');
```

24) XÂU CON

Cho xâu S độ dài n. Hãy tìm xâu con dài nhất của S, sao cho mỗi ký tự tham gia vào xâu con không quá k lần ($1 \leq n \leq 100\ 000$, $1 \leq k \leq n$).

Yêu cầu: Chỉ ra độ dài của xâu con tìm được và vị trí của ký tự đầu tiên thuộc xâu con trong xâu S ban đầu. Nếu có nhiều cách chọn xâu con – chỉ cần nêu một trong các cách.

Dữ liệu: Vào từ file văn bản SUBSTRG.INP:

- Dòng đầu tiên chứa 2 số nguyên n và k ,
- Dòng thứ hai chứa xâu S.

Kết quả: Đưa ra file văn bản SUBSTRG.OUT trên một dòng hai số nguyên: độ dài xâu con và vị trí ký tự đầu tiên của xâu con.

Ví dụ:

SUBSTRG.INP		SUBSTRG.OUT
5 2 ababa		4 2

Thuật toán:

<https://youtu.be/8nGH87osV2I>

- Tổ chức hàm kiểm tra xâu u có thỏa mãn: mỗi ký tự xuất hiện không quá K lần.
- **Ta có bài toán cơ sở:** Cho xâu, đếm số lần xuất hiện các ký tự trong xâu u.
- **Đếm phân phối** vào mảng C: array['A'..'z'] of longint; ch: char; t: string;

For i:= 1 to length(u) do

C[u[i]] := C[u[i]] + 1;

For ch:='A' to 'z' do

If C[ch] > 0 then writeln(ch,#32,c[ch]);

- Với mỗi xâu con của S, ta đếm phân phối lưu vào mảng C. Kiểm tra trong mảng C có phần tử C[ch] > k. Nếu có thì exit(false).

Exit(true).

```
Function kiemtra(u: string): boolean;
Var i: longint; C: array['A'..'z'] of longint;
Begin
    For i:= 1 to length(u) do
        C[u[i]] := C[u[i]] + 1;
    For ch:='A' to 'z' do
```

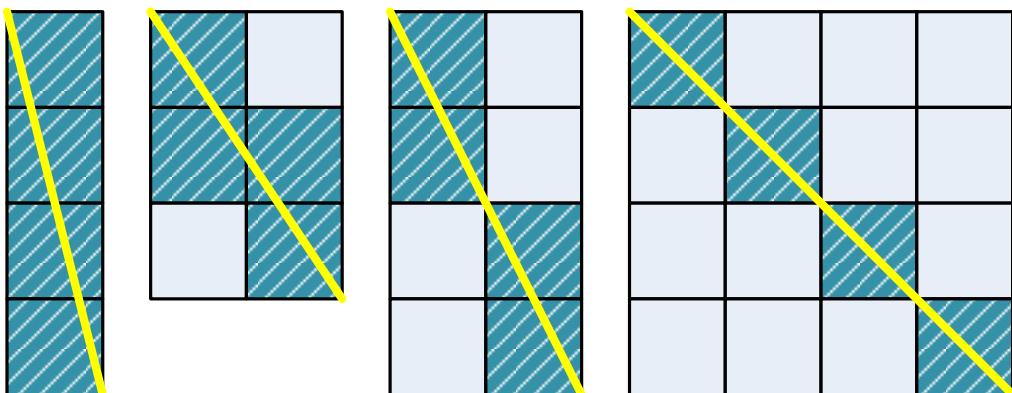
```

If C[ch] > k then exit(false);
Exit(true);
End;

Readln(n,k);
Readln(s);
Res := 0;
For i:=1 to length(s)-1 do
    For j:=i to length(s) do
        Begin
            T:= copy(S,i,j-i+1); // là xâu con từ i tới j
            If kiemtra(T) = true then
                Begin
                    If res < length(t) then
                        Begin
                            res := length(t);
                            i0:= i;
                        End;
                    End;
                End;
            End;
        Write(res, #32, i0);
    
```

25) Ô VUÔNG

Xét hình chữ nhật R kích thước nguyên $a \times b$. Hình chữ nhật này chứa $a \times b$ ô vuông đơn vị. Xét một đường chéo của R . Gọi $f(R)$ là số ô vuông của R có điểm trong chung với đường chéo đang xét. Ví dụ, với hình chữ nhật kích thước 2×4 , có $f(R) = 4$. Ngoài ra, còn có 3 hình chữ nhật khác cũng cho $f(R) = 4$, đó là các hình 1×4 , 3×2 và 4×4 (hai hình chữ nhật kích thước $a \times b$ và $b \times a$ được coi là giống nhau).



Yêu cầu: Cho số nguyên N ($0 < N < 10^6$). Hãy xác định số hình chữ nhật khác nhau có $f(R)=N$.

Dữ liệu: Vào từ file văn bản SQUARES.INP gồm một dòng chứa số nguyên N .

Kết quả: Đưa ra file văn bản SQUARES.OUT dưới dạng một số nguyên.

Ví dụ: // q47Jbalk08 5



SQUARES.INP		SQUARES.OUT
4		4

Thuật toán: <https://youtu.be/WKtdHsh0jQM>

- Tìm công thức tính số ô mà đường chéo chính đi qua của hình chữ nhật có cạnh a và b.
- Gọi $S = a + b - \text{ucln}(a,b)$.
- Cho N, hỏi có bao nhiêu bộ số (a, b) để $a + b - \text{ucln}(a,b) = N$ với $a \leq b$.
- Duyệt $(a, 1, N)$
Duyệt (b, a, N)
Nếu $a + b - \text{ucln}(a,b) = N$ thì $\text{res} := \text{res} + 1$;
- Đưa res ra.

```
1 #include <bits/stdc++.h>
2 using namespace std;
3 int main()
4 {
5     freopen("SQUARE.inp", "r", stdin);
6     freopen("SQUARE.out", "w", stdout);
7     int n;
8     cin >> n;
9     int res = 0;
10    for (int a = 1; a <= n; a++)
11        for (int b = a; b <= n; b++)
12            if (a + b - __gcd(a, b) == n) res += 1;
13    cout << res;
14    return 0;
15 }
16
```



Free Pascal IDE

```
[ ] File Edit Search Run Compile Debug Tools Option
var
  s, a, b, res, i, j, n: longint;
function ucln(u, v: longint): longint;
var r: longint;
begin
  while v > 0 do
    begin
      r := u mod v;
      u := v;
      v := r;
    end;
    exit(u);
end;
begin
  assign(input, 'squares.inp');reset(input);
  assign(output, 'squares.out');rewrite(output);
  readln(n);
  res := 0;
  for a := 1 to n do
    for b := a to n do
      if (a + b - ucln(a, b) = n) then inc(res);
  writeln(res);
  close(input);close(output);
end.
```

26) TỔNG CẶP SỐ

Xét dãy số nguyên dương khác nhau tùng đôi một a_1, a_2, \dots, a_n , trong đó $1 \leq a_i \leq 10^6$, $1 \leq n \leq 10^5$. Với số nguyên x cho trước ($1 \leq x \leq 200\,000$) hãy xác định số cặp (a_i, a_j) thỏa mãn các điều kiện:

- $a_i + a_j = x$,
- $1 \leq i < j \leq n$.

Dữ liệu: Vào từ file văn bản SUMX.INP:

- Dòng đầu tiên chứa số nguyên n ,
- Dòng thứ 2 chứa n số nguyên a_1, a_2, \dots, a_n ,
- Dòng thứ 3 chứa số nguyên x .

Kết quả: Đưa ra file văn bản SUMX.OUT một số nguyên – số cặp tìm được.

Ví dụ:

SUMX.INP	SUMX.OUT
9	3
5 12 7 10 9 1 2 3 11 13	

Thuật toán:



Subtask1: Duyệt (i, 1, N-1)

Duyệt (j, i + 1, N)

Nếu $a[i] + a[j] = x$ thì tăng res lên 1 đơn vị.

Đưa res ra.

Độ phức tạp: $O(N^2)$ với $N \leq 3000$.

Subtask2:

- Nhận xét: Chỉ được TKNP trên dãy tăng dần/ dãy tăng.
- Vì $a[i] + a[j] = x \rightarrow a[j] = x - a[i]$;
Duyệt (i, 1, N)
Nếu $TKNP(x - a[i]) > 0$ then $res := res + 1$;
- Sắp xếp dãy số thành dãy tăng (không giảm): Quicksort.

```
Procedure quicksort(L, H: longint);
Var i,j, chot, tam: longint;
Begin
  I:= L;
  J:= H;
  Chot:= a[(i+j) div 2];
  While i <= j do
    Begin
      While a[i] < chot do inc(i);
      While a[j] > chot do dec(j);
      If i <= j then
        Begin
          Tam:= a[i];
          A[i]:=a[j];
          A[j]:= tam;
          Inc(i); dec(j);
        End;
      End;
      If L < j then quicksort(L,j);
      If i < H then quicksort(i,H);
    End;
End;
```

Chương trình chính gọi: *Quicksort(I,N)*;

- Tổ chức hàm TKNP(u: longint): longint; trả về vị trí của u trong dãy a.

```
Function tknp(u: longint): longint;
Var dau, giua, cuoi: longint;
Begin
  Dau:=1; cuoi:= n;
  While dau <= cuoi do
    Begin
      Giua := (dau + cuoi) div 2;
```



```
If a[giua] = u then exit(giua);  
If a[giua] > u then cuoi := giua - 1  
Else dau := giua + 1;  
End;  
Exit(0);  
End;  
End;
```

Chương trình chính:

```
//Nhập dữ liệu  
quicksort(1,N);  
for i:=1 to n do  
    if tknp(x-a[i]) > 0 then inc(res);  
Đưa res ra.
```



```
[ ] var      i,j,chot,tam:longint;
begin
    i:=L;j:=H;
    chot:=a[(i+j) div 2];
    while i<=j do
    begin
        while a[i]<chot do inc(i);
        while a[j]>chot do dec(j);
        if i<=j then
        begin
            tam:=a[i];
            a[i]:=a[j];
            a[j]:=tam;
            inc(i);dec(j);
        end;
        if L<j then qsort(L,j);
        if i<H then qsort(i,H);
    end;
    function tknp(u:longint):longint;
var      dau,giua,cuoi:longint;
begin
    dau:=1; cuoi:=n;
    while dau<=cuoi do
    begin
        giua:=(dau+cuoi) div 2;
        if a[giua]=u then exit(giua);
        if a[giua]>u then cuoi:=giua-1
        else
            dau:=giua+1;
    end;
    exit(0);
end;
BEGIN
    assign(input,'bai22.INP');reset(input);
    assign(output,'bai22.OUT');rewrite(output);
    readln(n);
    for i:=1 to n do read(a[i]);
    readln(x);
    res:=0;
    qsort(1,n);
    for i:=1 to n div 2 do
        if tknp(x-a[i]) > 0 then inc(res);
    writeln(res);
    close(input);close(output);
END.
```



```
1 #include <bits/stdc++.h>
2 #define nmax 100007
3 using namespace std;
4 int a[nmax], n, x, res=0;
5 void sub1() //O(N^2)
6 {
7     for (int i=1;i<=n-1;i++)
8         for (int j=i+1;j<=n;j++)
9             if (a[i]+a[j]==x) res++;
10    cout<<res;
11 }
12 void qsort (int L, int H)
13 {
14     int i, j, chot, tam;
15     i=L;
16     j=H;
17     chot=a[(i+j)/2];
18     while (i<=j)
19     {
20         while (a[i]<chot) i++;
21         while (a[j]>chot) j--;
22     }
23     if (i<=j)
24     {
25         tam=a[i];
26         a[i]=a[j];
27         a[j]=tam;
28         i++;
29         j--;
30     }
31     if (L<j) qsort (L, j);
32     if (i<H) qsort (i, H);
33 }
34 int tknp (int u)
35 {
36     int dau=1, giua, cuoi=n;
37     while (dau<=cuoi)
38     {
39         giua=(dau+cuoi)/2;
40         if (a[giua]==u) return giua;
41         if (a[giua]>u) cuoi=giua-1;
42         else dau=giua+1;
43     }
44     return 0;
45 }
46 void sub2 ()
47 {
48     qsort (1, n);
49     for (int i=1;i<=n;i++)
50         if (tknp (x-a[i])>0)) res++;
51     cout<<res;
52 }
53 int main()
54 {
55     cin>>n;
56     for (int i=1;i<=n;i++)
57         cin>>a[i];
58     cin>>x;
59     if (n<=3000) sub1();
60     else sub2();
61     return 0;
62 }
```

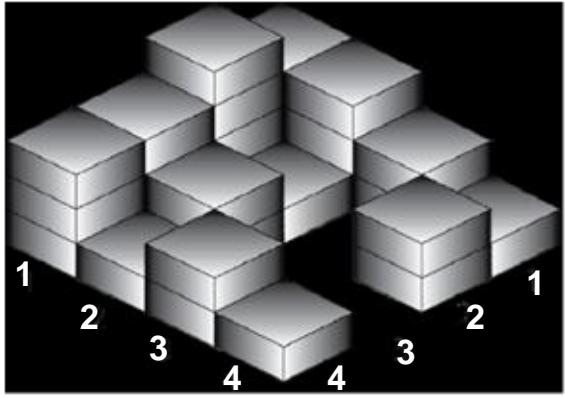


27) XẾP SÁCH

51



Thư viện trường vừa được bổ sung một khối lượng lớn sách. Người thủ thư phân loại sách và xếp chúng thành từng chồng trên một bàn cao. Mặt bàn được chia thành lưới $n \times n$ ô ($1 \leq n \leq 500$). Mỗi chồng sách chiếm vừa khít một ô. Có thể có các ô trống trên bàn. Các cuốn sách được dán dấu hiệu phân loại ở gáy và các phía xung quanh. Sinh viên trẻ nhất trường được giao nhiệm vụ ghi các sách mới vào phiếu tra cứu của thư viện. Ngán ngẫm nhìn đồng sách ngòn ngon, anh ta đi vòng quanh bàn, nhìn đồng sách theo các hướng song song với cạnh của bàn, đọc các phiếu phân loại của từng chồng nhìn thấy được. Một chồng sách có thể được nhìn thấy nếu giữa người sinh viên và chồng sách không có chồng nào cao hơn hoặc bằng theo hướng nhìn. Theo tinh huống ở hình bên, người sinh viên không thấy được chồng sách ở ô (2, 2). Các ô (2,3), (3, 3) và (3, 4) – ô trống.



Yêu cầu: Hãy xác định số chồng sách mà người sinh viên nhìn thấy được.

Dữ liệu: Vào từ file văn bản BOOKS.INP:

- Dòng đầu tiên chứa số nguyên n ,
- Dòng thứ i trong n dòng sau chứa n số nguyên xác định độ cao các chồng sách trong hàng, mỗi độ cao có giá trị không vượt quá 1 000.

Kết quả: Đưa ra file văn bản BOOKS.OUT một số nguyên – số chồng sách nhìn thấy được.

Ví dụ:

BOOKS .INP	BOOKS .OUT
4	12
3 3 2 1	
4 1 0 2	
3 2 0 0	
3 1 2 1	

28) NHÓM 3

Hãy tìm số
điều kiện:

lượng nhóm 3 số nguyên (i, j, k) thỏa mãn các

- $i = j \bmod 2$,
- $j = k \bmod 3$,
- $0 \leq i \leq j \leq k \leq n$.

Ví dụ, với $n = 4$ ta có 5 nhóm: (0, 0, 0), (0, 0, 3), (0, 2, 2), (1, 1, 1) và (1, 1, 4).

Dữ liệu: Vào từ file văn bản TRIPLES.INP gồm một dòng chứa số nguyên n ($1 \leq n \leq 100\,000$).

Kết quả: Đưa ra file văn bản TRIPLES.OUT một số nguyên – kết quả tìm được.

Ví dụ:

TRIPLES.INP		TRIPLES.OUT
4		5

Thuật toán: https://youtu.be/wf_7nBY-bWA

- Vì $i = j \bmod 2, j = k \bmod 3, 0 \leq i \leq j \leq k \leq n$



- For (k,0, N)

Begin

J:= k mod 3;

I:= j mod 2;

If (i <= j) and (j <= k) then inc(res);

End;

Đưa res ra.

```
[1] ===== triples.pas =====
var n,i,k,res,j:longint;
begin
  assign(input,'triples.inp'); reset(input);
  assign(output,'triples.out'); rewrite(output);
  readln(n);
  res:=0;
  for k:=0 to n do
    begin
      j:=k mod 3;
      i:=j mod 2;
      if (i<=j) and (j<=k) then inc(res);
    end;
  writeln(res);
  close(input);close(output);
end.
```

29) GIẢI MÃ

Các phương pháp mã hóa luôn có sức cuốn hút đặc biệt đối với Rôn. Xuất phát từ việc mọi thông tin đều được lưu trữ dưới dạng số, Rôn nghĩ rằng chỉ cần phát triển các phương pháp mã hóa số nguyên. Mới đây Rôn đề xuất một phương pháp mã hóa của riêng mình: mỗi số nguyên x được Rôn mã hóa thành số nguyên y bằng cách *cộng vào x các chữ số của nó* (ở hệ thập phân). Như vậy, nếu $x = 12$, ta sẽ có $y = 12 + 1 + 2 = 15$.

Mã hóa bao giờ cũng đi đôi với việc giải mã. Biết $y = 15$, ta phải tìm được số ban đầu $x = 12$.

Yêu cầu: Cho số nguyên dương y . Hãy xác định số ban đầu chưa được mã hóa. Dữ liệu đảm bảo có kết quả giải mã.

Dữ liệu: Vào từ file văn bản DECODE.INP gồm một dòng chứa số nguyên y ($1 \leq y \leq 10^9$).

Kết quả: Đưa ra file văn bản DECODE.OUT một số nguyên – kết quả giải mã.

Ví dụ:

DECODE.INP	DECODE.OUT
15	12

Thuật toán:

<https://youtu.be/-9coMjF8kD4>

Giải mã $Y = x + \text{tongcs}(x)$

Duyệt tất cả các x có giá trị trong đoạn $[1, y]$ sao cho thoả mãn điều kiện $Y = x + \text{tongcs}(x)$ thì đưa x ra và break.

- Tạo chức hàm $\text{tongcs}(u: \text{longint}): \text{longint}$;

Function $\text{tongcs}(u: \text{longint}): \text{longint}$;

Var s: longint = 0;

Begin

While $u > 0$ do



```
Begin
    S:= s+ u mod 10;
    U:= u div 10;
End;
exit(s);
End;

//đọc dữ liệu
P:= 0;
For i:= 1 to y do
    Nếu tongcs(i) + i = y thì
        Begin
            Write(i);
            P:=1;
            Break;
        End;
    If p = 0 then write(-1);
```



```
[■]
Var y,i,p: longint;
Function Tongcs(x: longint): longint;
Var tong: longint;
Begin
    Tong:=0;
    While x>0 do
        Begin
            tong:=tong+(x mod 10);
            x:=x div 10;
        End;
    Exit(tong);
End;
Begin
    Assign(Input,'DAY10_BAI2.INP');
    Assign(Output,'DAY10_BAI2.OUT');
    Reset(Input);
    Rewrite(Output);
    Readln(y);
    P:=0;
    For i:=1 to y do
        If Tongcs(i)+i=y then
            Begin
                Write(i);
                P:=1;
                Break;
            End;
    If p=0 then write('-1');
    Close(Input);
    Close(Output);
End.
```

30) DÃY SỐ ĐẶC BIỆT

Xét các số nguyên dương mà trong dạng biểu diễn thập phân có bình phương chỉ chứa các chữ số 0, 4 và 9, mỗi chữ số này xuất hiện ít nhất một lần. Những số này được gọi là số đặc biệt. Ví dụ, 2120 là một số đặc biệt vì $2120^2 = 4494400$. Một số đặc biệt khác là 97 vì $97^2 = 9409$. Các số 13 và 7 không phải là những số đặc biệt vì $13^2 = 169$ – chúa các chữ số ngoài tập đang xét, $7^2 = 49$ – thiếu chữ số 0. Xét tập sắp xếp theo thứ tự tăng dần các số đặc biệt { 70, 97, 700, 970, 997, 2120, 3148, 7000, 9700, 9970, 9997, 20102, 21200, 31480, 70000, 97000,... } Các số được đánh thứ tự từ 1 trở đi.

Yêu cầu: Cho số nguyên n ($1 \leq n \leq 250$). Hãy xác định số đặc biệt thứ n .

Dữ liệu: Vào từ file văn bản SPECIAL.INP gồm một dòng chứa số nguyên n .

Kết quả: Đưa ra file văn bản SPECIAL.OUT số đặc biệt tìm được.

Ví dụ:

SPECIAL.INP	SPECIAL.OUT
12	20102

Thuật toán: *Bài toán tìm số thứ N của mảng a[].*

<https://youtu.be/OKXpMabekcg>



Function kiemtra(u: longint): boolean; //kiểm tra u đồng thời và chỉ chứa 0, 4, 9.

Var d1,d2,d3: longint;

Begin

 D1:=0; d2:=0;

 d3:=0;

 while u > 0 do

 Begin

 P:= U mod 10;

 If p = 0 then d1:=1;

 If p = 4 then d2:=1;

 If p = 9 then d3 := 1;

 If not((p=0) or (p=4) or (p=9)) then exit(false);

 U:= u div 10;

 End;

 If d1+d2+d3 = 3 then exit(true)

 Else exit(false);

 End;

Function timson(u: longint): Longint;

Var

Begin

 A[0]:= 0; i:=1;

 While i<=u do

 Begin

 P:= a[i-1] + 1;

 While kiemtra(p*p) = false do inc(p);

 A[i]:= p;

 Inc(i);

 End;

 Exit(a[u]);

 End;

//đọc dữ liệu

Write(timson(n));



```
[ ] ===== DAY10_BAI3.pas =====  
Var n: longint;  
    a:array[-1..100000] of longint;  
Function kiemtra(u: longint): boolean;  
Var d1,d2,d3,p: longint;  
Begin  
    d1:=0; d2:=0; d3:=0;  
    While u>0 do  
        Begin  
            p:=u mod 10;  
            If p=0 then d1:=1;  
            If p=4 then d2:=1;  
            If p=9 then d3:=1;  
            If not ((p=0) or (p=4) or (p=9)) then exit(false);  
            u:= u div 10;  
        End;  
    If d1+d2+d3=3 then exit(true)  
    Else exit(false);  
End;  
Function Timson(u: longint): longint;  
Var i,p: longint;  
Begin  
    A[0]:=0;  
    i:=1;  
    While i<=u do  
        Begin  
            p:=a[i-1]+1;  
            While kiemtra(p*p)= false do inc(p);  
            A[i]:=p;  
            Inc(i);  
        End;  
    Exit(a[u]);  
End;  
Begin  
    Assign(Input, 'DAY10_BAI3.INP');  
    Assign(Output, 'DAY10_BAI3.OUT');  
    Reset(Input);  
    Rewrite(Output);  
    ReadLn(n);  
    Write(Timson(n));  
    Close(Input);  
    Close(Output);  
End.
```



31) XÂU FIBONACCI

Công thức lặp có thể gặp với cả biểu thức xâu. Biểu thức xâu Fibonacci được xác định bằng công thức lặp $F_0 = a$, $F_1 = b$, $F_2 = F_0 + F_1$, ..., $F_n = F_{n-2} + F_{n-1}$, ... Các xâu đầu tiên xác định theo công thức lặp này là $a, b, ab, bab, abbab, bababbab, abbabbababbab, \dots$. Độ dài của xâu tăng lên rất nhanh. Vì vậy ta chỉ xét bài toán xác định một ký tự của một xâu trong dãy các xâu này.

Yêu cầu: Cho 2 số nguyên n và k . Hãy xác định ký tự thứ k của xâu F_n . Các ký tự trong F_n được đánh số bắt đầu từ 1.

Dữ liệu: Vào từ file văn bản FIB1.INP:

- Dòng đầu tiên chứa số nguyên T – số bộ dữ liệu test ($1 \leq T \leq 100$),
- Mỗi dòng sau chứa 2 số nguyên n và k ($0 \leq n \leq 45$, $1 \leq k \leq \text{length}(F_n)$).

Kết quả: Đưa ra file văn bản FIB1.OUT, kết quả mỗi test đưa ra trên một dòng dưới dạng một ký tự.

Ví dụ:

FIB1.INP	FIB1.OUT
4	a
0 1	b
1 1	a
3 2	a
7 7	

Thuật toán: <https://youtu.be/-TCnbCq5l8I>

- Multiple Test
- Thay vì với mỗi N ta tính lại các $F[1], F[2], \dots, F[n]$.
- Ta tìm N_{\max} và tính tới $F[n_{\max}]$ một lần duy nhất.

```
F: array[0..100000] of string;
```

```
Readln(T);
```

```
For i:=1 to T do
```

```
Begin
```

```
    Readln(a[i], b[i]);
```

```
    Res := max(res, a[i]);
```

```
End;
```

```
Tìm số Fibonacci lớn nhất
```

```
F[0] := 'a'; F[1]:= 'b';
```

```
For i:= 2 to res do f[i] := f[i-1] + f[i-2];
```

```
Với mỗi test ta thực hiện: Writeln(f[a[i]][b[i]]);
```

32) MUÒI MỘT

Hãy xác định số lượng các số nguyên chia hết cho 11 trong khoảng $[A, B]$ và ở dạng biểu diễn hệ thập phân có tổng các chữ số nằm trong khoảng $[P, Q]$.

Dữ liệu: Vào từ file văn bản ELEVEN.INP:

- Dòng đầu tiên chứa 2 số nguyên A, B ,
- Dòng thứ 2 chứa 2 số nguyên P, Q .

Kết quả: Đưa ra file văn bản ELEVEN.OUT số lượng tìm được.



Ví dụ:



ELEVEN.INP	ELEVEN.OUT
11 40	2
3 6	

Thuật toán: <https://youtu.be/ylfVKtW0JKg>

Subtask1:

- Độ phức tạp $O(b \cdot \log(b))$ với $b \leq 10^5$.
- Hàm tính tổng chữ số của u: function tongcs(u:longint): longint;

For (i, a, b)

Begin

 k:= tongcs(i);
 Nếu ($i \bmod 11 = 0$) and ($k \geq p$) and ($k \leq q$) thì inc(res);

End;

```
var  
    a,b,p,q,i,dem:longint;  
function tongcs(u:longint):longint;  
var  
    s:longint;  
begin  
    s:=0;  
    while u>0 do  
    begin  
        s:=s+u mod 10;  
        u:=u div 10;  
    end;  
    exit(s);  
end;  
BEGIN  
    assign(input,'ELEVEN.INP');reset(input);  
    assign(output,'ELEVEN.OUT');rewrite(output);  
    readln(a,b);  
    readln(p,q);  
    for i:=a to b do  
        if (i mod 11=0) and (tongcs(i)>=p) and (tongcs(i)<=q) then inc(dem);  
    write(dem);  
    close(input);close(output);  
END.
```



```
[ ] ----- ELEVEN.pas -----
var s: longint;
Begin
    S:=0;
    While u>0 do
    Begin
        S:=s+u mod 10;
        u:=u div 10;
    End;
    Exit(s);
End;
Begin
    Assign(Input,'ELEVEN.INP');
    Assign(Output,'ELEVEN.OUT');
    Reset(Input);
    Rewrite(Output);
    Readln(a,b);
    Readln(p,q);
    For i:=a to b do
        Begin
            k:=tongcs(i);
            If (i mod 11=0) and (k>=p) and (k<=q) then inc(dem);
        End;
    Write(dem);
    Close(Input);
    Close(Output);
End.
```

Subtask2:

- Tam := a div 11;
- Nếu a mod 11 <>0 thì tam:=tam+1;

Khoảng cần kiểm tra i thuộc đoạn [11*tam, b]

- i:= 11*tam; res := 0;
- while i <=b do
 - begin
 - k:= tongcs(i);
 - if (k >= p) và (k <=q) then inc(res);
 - inc(i,11);
- end;

- Đưa res ra.

```
[ ] -----
var
    a, b, p, q, d, i, k, tam: longint;
function tcs(u: longint): longint;
var tam: longint;
begin
    tam := 0;
    while u > 0 do
        begin
            tam := tam + (u mod 10);
            u := u div 10;
        end;
    exit(tam);
end;
```



```
begin
    assign(input,'eleven.inp');reset(input);
    assign(output,'eleven.out');rewrite(output);
    readln(a, b);
    readln(p, q);
    d := 0;tam := a div 11;
    if a mod 11 <> 0 then tam := tam + 1;
    i := tam * 11;
    while i < b do
        begin
            k := tcs(i);
            if (p <= k) and (k <= q) then inc(d);
            inc(i,11);
        end;
    writeln(d);
    close(input);close(output);
end.
```



```
1 #include <bits/stdc++.h>
2 using namespace std;
3 int a,b,q,p;
4 int tongcs(int x)
5 {
6     int t=0;
7     while(x!=0)
8     {
9         t=t+(x%10);
10        x=x/10;
11    }
12    return t;
13 }
14 void sub1() //b<=10^5
15 {
16     int res=0;
17     for(int i=a;i<=b;i++)
18     {
19         if(i%11==0)
20         {
21             int k=tongcs(i);
22             if(k>=p&&k<=q)
23                 res++;
24         }
25     cout<<res;
26 }
27 void sub2()
28 {
29     int res=0,k;
30     int tam=a/11;
31     if(a%11!=0) tam++;
32     int i=tam*11;
33     while(i<=b)
34     {
35         k=tongcs(i);
36         if(p<=k&&k<=q) res+=1;
37         k=tongcs(i);
38         if(p<=k&&k<=q) res+=1;
39         i+=11;
40     }
41     cout<<res;
42 }
43 int main()
44 {
45     cin>>a>>b>>p>>q;
46     if(b<=100000) sub1();
47     else sub2();
48     return 0;
49 }
```

33) CHỮ SỐ HÀNG ĐƠN VỊ

Cho 3 số nguyên a, k, b ($0 \leq a \leq 1\,000, 1 \leq b, k \leq 1\,000$). Dãy số nguyên được xây dựng theo quy tắc sau: số đầu tiên là a , các số sau được xây dựng từ số trước theo quy tắc: gọi số cuối cùng trong dãy đã xây dựng là r , nếu r chia hết cho $(k+b)$ thì số tiếp theo là $(k*r+1)$, trong trường hợp ngược lại – là $(b*r+2)$.

Yêu cầu: Cho số nguyên N ($1 \leq N \leq 2*10^9$). Hãy xác định chữ số hàng đơn vị của phần tử thứ N của dãy.

Dữ liệu: Vào từ file văn bản DIGIT.INP:

- Dòng đầu tiên chứa 3 số nguyên a, k, b ,
- Dòng thứ 2 chứa số nguyên N .

Kết quả: Đưa ra file văn bản DIGIT.OUT chữ số tìm được.

Ví dụ:



DIGIT.INP	DIGIT.OUT
1 2 3	1
5	

Thuật toán: <https://youtu.be/KUji0Mgai1U>

- Gọi c[i] là mảng cần xây dựng.

C[1] := a;

For (i,2, N)

Nếu c[i-1] mod (k+b)=0 thì c[i] := k*c[i-1] + 1

Ngược lại c[i]:= b*c[i-1] + 2;

- Write(c[n] mod 10);

```

Free Pascal IDE
File Edit Search Run Compile Debug Tools Options Windows
[ ] - var
      a,b,k,n,i,r:longint;
      c:array[1..100000] of longint;
BEGIN
  assign(input,'DIGIT.INP');reset(input);
  assign(output,'DIGIT.OUT');rewrite(output);
  readln(a,k,b);
  readln(n);
  c[1]:=a;
  for i:=2 to n do
    if c[i-1] mod (k+b)=0 then c[i]:=k*r+1
    else
      c[i]:=b*r+2;
  writeln(c[n] mod 10);
  close(input);close(output);
END.

```

```

1 #include <bits/stdc++.h>
2 #define nmax 100007
3 using namespace std;
4 int a,k,b,n,c[nmax];
5 void subbruh()
6 {
7     c[1]=a;
8     for(int i=2;i<=n;i++)
9         if(c[i-1]%(k+b)==0)
10             c[i]=k*c[i-1]+1;
11         else c[i]=k+c[i-1]+2;
12     cout<<c[n]%10;
13 }
14 int main()
15 {
16     cin>>a>>k>>b;
17     cin>>n;
18     subbruh();
19     return 0;
20 }
21

```

34) SỐ ĐỐI XỨNG

Số đối xứng là số có thể viết từ trái sang phải các chữ số của nó ta vẫn được chính nó. Từ một số có hai chữ số ta có thể nhận được một số đối xứng theo cách sau: lấy số ban đầu cộng với số ánh xạ



gương của nó, tức là số nhận được bằng cách đọc các chữ số từ phải sang trái. Nếu chưa phải là số đối xứng, số đó lại được cộng với ánh xạ gương của nó và tiếp tục như vậy cho đến khi nhận được số đối xứng. Ví dụ, từ số 48 ta có $48+84 = 132$, $132+231 = 363$. Như vậy 48 tương ứng với 363.

Yêu cầu: Tìm số đối xứng của N ($11 \leq N \leq 99$).

Dữ liệu: Vào từ file văn bản PALNUM.INP chứa số nguyên N.

Kết quả: Đưa ra file văn bản PALNUM.OUT số đối xứng tương ứng.

Ví dụ:

PALNUM.INP	PALNUM.OUT
48	363

Thuật toán: <https://youtu.be/l4iR4NDAsMI>

- Hàm tìm số đảo của số u: function sodao(u: longint): longint;
- Hàm kiểm tra tính đối xứng của u: function palin(u: longint) :boolean;
- Chương trình chính:
N:= n + sodao(n);
While pali(n) = false do
 N := n + sodao(n);
Write(n);

```
Free Pascal IDE
File Edit Search Run Compile Debug Tools Options Windows
[ ]
var
  n,somoi:longint;
function sodao(x:longint):longint;
var
  s:longint;
begin
  s:=0;
  while x>0 do
  begin
    s:=s*10+x mod 10;
    x:=x div 10;
  end;
  exit(s);
end;
function sodoxung(x:longint):boolean;
begin
  if x=sodao(x) then exit(true) else exit(false);
end;
BEGIN
  assign(input,'PALNUM.INP');reset(input);
  assign(output,'PALNUM.OUT');rewrite(output);
  readln(n);
  n:=n+sodao(n);
  while sodoxung(n)=false do n:=n+sodao(n);
  write(n);
  close(input);close(output);
END.
```



```
1 #include <bits/stdc++.h>
2 using namespace std;
3 int n,p=0;
4 bool palnum(int u)
5 {
6     int tam=0, v=u;
7     while(u>0)
8     {
9         tam=tam*10+(u%10);
10        u=u/10;
11    }
12    if(tam==v) return true;
13    else return false;
14 }
15 int timsdao(int u)
16 {
17     int tam=0;
18     while(u>0)
19     {
20         tam=tam*10+(u%10);
21         u=u/10;
22     }
23     return tam;
24 }
25 void subpalinum()
26 {
27     int m=n+timsdao(n);
28     while(!palnum(m))
29     {
30         m+=timsdao(m);
31         p=1;
32     }
33     if(p==1) cout<<m;
34     else cout<<-1;
35 }
36 int main()
37 {
```



Free Pascal IDE

```

File Edit Search Run Compile Debug Tool
[ ] palnum.pas
uses math;
var n,i:longint;

function doixung(u:longint):boolean;
var j,k:longint;
    s:string;
begin
    str(u,s);
    k:=length(s);
    for j:= 1 to (k div 2) do
        begin
            if s[j] <> s[k-j+1] then
                exit (false);
        end;
    exit(true);
end;
function sodao(u:longint):longint;
var tam,s:longint;
begin
    s:=0;
    while u>0 do
        begin
            s:=s*10+(u mod 10);
            u:=u div 10;
        end;
    exit(s);
end;
begin
    assign(input,'palnum.inp'); reset(input);
    assign(output,'palnum.out'); rewrite(output);
    readln(n);
    n:=n+sodao(n);
    while doixung(n)=false do
        n:=n+sodao(n);
    writeln(n);
    close(input);close(output);
end.

```

38:1 =
F1 Help F2 Save F3 Open Alt+F9 Compile F9 Make Alt+F1

35) GIẢI MÃ SỐ

Các chữ số từ 1 đến 9 được mã hoá dưới dạng các từ chỉ chứa các ký tự a, b và c theo quy tắc sau:

Chữ số	1	2	3	4	5	6	7	8	9
Mã	a	b	cc	bbc	cbc	abc	bac	aac	cac

Ví dụ số 132 sẽ được viết thành accb.

abcaa

Yêu cầu: từ xâu ký tự cho trước, hãy tìm số nguyên dương tương ứng.

Dữ liệu: Vào từ file văn bản DECODE.INP xâu chứa không quá 100 ký tự a,b, c.

Kết quả: Đưa ra file văn bản DECODE.OUT số tương ứng hoặc -1 nếu xâu không tương ứng với một số nguyên nào.

Ví dụ:

DECODE.INP		DECODE.OUT
abcac		129

Thuật toán: <https://youtu.be/w8Iaqx8xshw>

- Tách các xâu con có độ dài i (i thuộc đoạn [1,3]).
- Với mỗi xâu con ta kiểm tra xâu có xuất hiện trong **Mã**. Nếu có, ghi nhớ cách tách.
- Xâu còn lại có độ dài L – i + 1;
- Lặp lại đến khi xâu S rỗng.



- Sau khi thực hiện tách như trên, mà xâu S vẫn còn thì phép tách không đúng và ta thử phép tách kế tiếp.



36) SỐ MỘT

Xét các số nguyên dương K có dạng biểu diễn ở hệ thập phân có N chữ số và chỉ bao gồm các chữ số 1, ví dụ với N = 2 có K = 11, N = 3 có K = 111.

Yêu cầu: Với N cho trước, tính K^2 ($1 \leq N \leq 10^5$)

Dữ liệu: Vào từ file văn bản ONES.INP chứa một số nguyên N.

Kết quả: Đưa ra file văn bản ONES.OUT giá trị K^2 tương ứng chia dư cho 10^9 .

Ví dụ:

ONES.INP	ONES.OUT
9	987654321

Thuật toán:

<https://youtu.be/x09l7Yab2Qk>

- Tô chức hàm tìm số có u chữ số 1:

```
Function timson(u: longint):longint;
Var s, d: longint;
Begin
  S:= 1;
  D:= 1;
  While d < u do
    Begin
      S:= S*10 + 1;
      Inc(d);
    End;
  Exit(s);
End;
```

Chương trình chính

```
P:= timson(n);
Write(p*p mod trunc(1e9));
```

```
[ ]——
Var p,n: Int64;
Function timson(u: Int64): Int64;
Var s,d: Int64;
Begin
  S:=1;
  D:=1;
  While d<u do
    Begin
      S:=S*10+1;
      Inc(d);
    End;
  Exit(s);
End;
Begin
  Assign(Input,'ONES.INP');
  Assign(Output,'ONES.OUT');
  Reset(Input);
  Rewrite(Output);
  Readln(n);
  P:=timson(n);
  Write(p*p mod trunc(1e9));
  Close(Input);
  Close(Output);
End.
```



Free Pascal IDE

```
File Edit Search Run Compile Debug Tools Options  
[ ]  
var  
  n, i: longint;  
  k: int64;  
function ts(u: longint): longint;  
var s, d: longint;  
begin  
  s := 1; d := 1;  
  while d <= u do  
  begin  
    s := s * 10 + 1;  
    inc(d);  
  end;  
  exit(d);  
end;  
begin  
  assign(input, 'ones.inp'); reset(input);  
  assign(output, 'ones.out'); rewrite(output);  
  readln(n);  
  k := 1; // k := ts(n);  
  for i := 2 to n do k := k * 10 + 1;  
  writeln((k*k) mod (trunc(1e9)));  
  close(input); close(output);  
end.  
  
10:60 ←  
F1 Help F2 Save F3 Open Alt+F9 Compile F9 Make
```

```
1 #include <bits/stdc++.h>  
2 using namespace std;  
3 int n, p=0;  
4 int timso(int u)  
5 {  
6     int s=1, d=1;  
7     while (d<u)  
8     {  
9         s=s*10+1;  
10        d++;  
11    }  
12    return s;  
13}  
14 int main()  
15 {  
16     cin>>n;  
17     int k=timso(n);  
18     //for(int i=2;i<=n;i++) k=k*10+1;  
19     cout<<(k*k)*111;  
20     return 0;  
21 }  
22 }
```

37) CÁC NỀN VĂN MINH CỔ ĐẠI

Các nền văn minh cổ đại có những ảnh hưởng qua lại với nhau trực tiếp hoặc gián tiếp nếu có khoảng thời gian cùng tồn tại. Ảnh hưởng sẽ càng lớn nếu thời gian tồn tại đồng thời lớn. Ví dụ, nếu nền văn minh A tồn tại từ 600 năm trước công nguyên (CN) đến trước năm 400 trước CN, còn nền văn minh B xuất hiện vào năm 450 trước CN và tồn tại đến trước năm 300 trước CN, thì giữa chúng có 50 năm cùng tồn tại. Nếu nền văn minh C xuất hiện vào năm 400 trước CN và tồn tại đến trước năm 50 trước



CN thì giữa A và C không có tác động qua lại, trong lúc đó giữa B và C có đến 100 năm cùng tồn tại và ảnh hưởng qua lại.

Để đánh giá mức độ ảnh hưởng theo thời gian, các nhà khảo cổ học quyết định chọn 2 nền văn minh có khoảng thời gian cùng tồn tại khác 0 nhỏ nhất trong số N nền văn minh mà tài liệu còn được ghi chép và tìm thấy ($1 \leq N \leq 100\,000$).

Dữ liệu: Vào từ file văn bản ANCIENT.INP:

- Dòng đầu tiên chứa số nguyên N,
- N dòng sau mỗi dòng chứa 2 số nguyên S_i và E_i , trong đó S_i – năm xuất hiện, E_i – năm diệt vong, giá trị âm hoặc 0 thể hiện trước công nguyên, giá trị tuyệt đối của năm không vượt quá 10^9 .

Kết quả: Đưa ra file văn bản ANCIENT.OUT gồm:

Dòng 1: Ghi thời gian cùng tồn tại nhỏ nhất của hai nền văn minh.

Dòng 2: Ghi hai số nguyên trên một dòng xác định các nền văn minh có thời gian cùng tồn tại khác 0 nhỏ nhất. Nếu không tìm thấy 2 nền văn minh nào cùng tồn tại thì đưa ra một số 0.

Ví dụ:

ANCIENT.INP	ANCIENT.OUT
3	1 3
-10 80	
-80 10	
60 90	

Thuật toán:

<https://youtu.be/vEwSBMNiuI8>

- Nhập N.
- For($i, 1, n$) readln($S[i], E[i]$);
- Sắp xếp các khoảng thời gian tăng dần theo năm bắt đầu $s[i]$. Khi đổi chỗ $S[i]$ cho $S[j]$ thì ta cũng đổi chỗ $E[i]$ cho $E[j]$.
- Ta nhận thấy nền văn minh thứ i có giao nhau với nền văn minh thứ $i+1$ khi:
 - $S[i+1] \leq E[i]$: nếu $e[i] < e[i+1]$ thì $res := \min(res, abs(s[i+1] - e[i]))$
Ngược lại, nếu $e[i] > e[i+1]$ thì $res := \min(res, abs(s[i+1] - e[i+1]))$.
- Duyệt ($i, 1, N-1$)
 - $S[i+1] \leq E[i]$: nếu $e[i] < e[i+1]$ thì $res := \min(res, abs(s[i+1] - e[i]))$
Ngược lại, nếu $e[i] > e[i+1]$ thì $res := \min(res, abs(s[i+1] - e[i+1]))$.
- Đưa res ra.

38) TỔNG SỐ NGUYÊN TỐ 1

Cho số nguyên dương N. Hãy đếm số cách phân tích số N thành tổng các số nguyên tố khác nhau.

Với $N = 5$, tìm được 7 cách phân tích N như sau:

$$\begin{aligned}
 5 &= 1 + 1 + 1 + 1 + 1 \\
 &= 1 + 1 + 1 + 2 \\
 &= 1 + 1 + 3 \\
 &= 1 + 2 + 2 \\
 &= 1 + 4 \\
 &= 2 + 3
 \end{aligned}$$

Trong đó, có 1 cách phân tích $5 = 2 + 3$ là các số nguyên tố khác nhau



Yêu cầu: Hãy đếm số cách phân tích số N thành tổng các số nguyên tố khác nhau.

Dữ liệu vào: Từ tệp văn bản PTPRIME2.INP chứa số nguyên dương N ($N \leq 10^6$).

Dữ liệu ra: Ghi ra tệp văn bản PTPRIME.OUT số cách phân tích số N thành tổng các số nguyên tố khác nhau.

PTPRIME .INP
5

PTPRIME .OUT
1

Thuật toán:

- Hàm kiểm tra mảng a có u phần tử chỉ chứa các số nguyên tố khác nhau hay không?
- Thủ tục phân tích số nguyên dương u thành tổng các số nguyên dương.
- Cải tiến thủ tục xử lí (u: longint): Kiểm tra mảng a chỉ chứa các số nguyên tố khác nhau hay không?

Vì dãy a[i] là dãy tăng dần, kiểm tra a[i] có thỏa mãn điều kiện.

```
Function nguyento(u: longint) : boolean; //Hàm nguyên tố
```

```
....
```

```
Procedure xuli(i: longint); //thủ tục xử lí mỗi câu hình
```

```
Var
```

```
    J: longint;
```

```
Begin
```

```
    P:=1;
```

```
    For j:=1 to i do
```

```
        If nguyento(a[j]) = false then
```

```
            Begin
```

```
                P:=0; break;
```

```
            End;
```

```
        If p = 1 then
```

```
            Begin
```

```
                For j:=2 to i do
```

```
                    If a[j-1] = a[j] then
```

```
                        Begin
```

```
                            P:= 0;
```

```
                            Break;
```

```
                        End;
```

```
                    If p = 1 then
```

```
                        Begin
```

```
                            For j:=1 to i do write(a[j],#32);
```

```
                            Writeln();
```

```
                        End;
```

```
                    End;
```

```
    End;
```

```
Procedure xuli(u: longint);
```



```
Var i, p : longint;
Begin
  P:=1;
  For i:= 2 to u do
    if not ((nguyento(a[i]) = true) and (a[i] <> a[i-1])) then
      begin
        p:=0;
        break;
      end;
  if p=1 then
    begin
      for i:=1 to u do write(a[i],#32);
      writeln();
    end;
End;
```

```
[ ] kk.pas
uses math;
var
  a,t: array [-5..100007] of longint;
  n,i: longint;
function nt(k: longint): boolean;
var j: longint;
begin
  if k <= 1 then exit(false);
  if k <= 3 then exit(true);
  for j:= 2 to trunc(sqrt(k)) do
    if k mod j = 0 then exit(false);
  exit(true);
end;
procedure xuli(k: longint);
var p,i: longint;
begin
  p:= 1;
  for i:= 2 to k do
    if not((nt(a[i]) = true) and ((a[i]) <> a[i - 1])) then
      begin
        p:= 0;
        break;
      end;
  if p = 1 then
    begin
      for i:= 1 to k do write(a[i],#32);
      writeln();
    end;
end;
```



```
procedure thu(i: longint);
var j: longint;
begin
    for j:= a[i + 1] to (n - t[i + 1]) div 2 do
        begin
            a[i]:= j;
            t[i]:= t[i + 1] + j;
            thu(i + 1);
        end;
    a[i]:= n - t[i + 1];
    xuli(i);
end;
begin
    assign(input,'kk.inp'); reset(input);
    assign(output,'kk.out'); rewrite(output);
    readln(n);
    a[0]:= 1; t[0]:= 0;
    thu(1);
    close(input);
    close(output);
end.
```

25:19



```
1 #include<bits/stdc++.h>
2 #define nmax 32
3 using namespace std;
4 int a[nmax], t[nmax], n, f[100007];
5 bool snt(int u)
6 {
7     if(u<=1) return false;
8     if(u==2 || u==3) return true;
9     for(int i=2; i<=trunc(sqrt(u)); i++)
10         if(u%i==0) return false;
11     return true;
12 }
13 void xl(int u)
14 {
15     int p=1;
16     for(int i=2; i<=u; i++)
17         if(!(snt(a[i]) && a[i] != a[i-1]))
18         {
19             p=0;
20             if(u<=1) return false;
21             if(u==2 || u==3) return true;
22             for(int i=2; i<=trunc(sqrt(u)); i++)
23                 if(u%i==0) return false;
24             return true;
25         }
26     void xl(int u)
27     {
28         int p=1;
29         for(int i=2; i<=u; i++)
30             if(!(snt(a[i]) && a[i] != a[i-1]))
31             {
32                 p=0;
33                 break;
34             }
35         if(p==1)
36         {
37             for(int i=1; i<=u; i++) cout<<a[i]<<" ";
38             cout<<"\n";
39         }
40     }
41     int main()
42     {
43         cin>>n;
44         a[0]=1; t[0]=0;
45         thu(1);
46         return 0;
47     }
```



39) TỔNG SỐ NGUYÊN TỐ 2

Cho số nguyên dương N. Trong các cách phân tích số N thành tổng số nguyên tố khác nhau. Đưa ra một cách phân tích chỉ gồm số nguyên tố khác nhau và nhiều số nhất.

Với $N = 5$, tìm được 7 cách phân tích N như sau:

$$\begin{aligned}5 &= 1 + 1 + 1 + 1 + 1 \\&= 1 + 1 + 1 + 2 \\&= 1 + 1 + 3 \\&= 1 + 2 + 2 \\&= 1 + 4 \\&= 2 + 3 \\&= 5\end{aligned}$$

Trong đó, có 1 cách phân tích $5 = 2 + 3$ là các số nguyên tố khác nhau và nhiều nhất là 2 số nguyên tố.

Yêu cầu: Hãy đếm số cách phân tích số N thành tổng các số nguyên tố khác nhau và có số lượng số nguyên tố là nhiều nhất.

Dữ liệu vào: Từ tệp văn bản PTPRIME2.INP chứa số nguyên dương N ($N \leq 10^6$).

Dữ liệu ra: Ghi ra tệp văn bản PTPRIME2.OUT kết quả tính được.

PTPRIME2.INP	PTPRIME2.OUT
5	2 2 3

Thuật toán:

- Thủ tục phân tích số nguyên dương u thành tổng các số nguyên dương.
- Cải tiến thủ tục xử lí (u: longint): Kiểm tra mảng a chỉ chứa các số nguyên tố khác nhau hay không?

Nếu có, cập nhật số lượng phần tử nhiều nhất của a.

Vì dãy a[i] là dãy tăng dần, kiểm tra a[i] có thỏa mãn điều kiện.



```
Procedure xuli(u: longint);
Var
J: longint;
Begin
P:= 1; //lưu trạng thái dãy có thoả mãn điều kiện
For i:=1 to u do
Begin
If a[i] = a[i-1] then p:= 0; //có hai phần tử bằng nhau
If nguyento(a[i]) = false then p:=0; //có phần tử không là số nguyên tố
If p = 0 then break;
End;
If p = 1 then
Begin
If res < u then
Begin
Res := u;
For i:=1 to u do d[i] := a[i]; //lưu cấu hình vào d[i] có res phần tử
End;
End;
End;
Chuong trình chính:
A[0]:=1; T[0]:=0; res := 0;
Thu(1);
Writeln(res);
For i:= 1 to res do write(d[i],#32); //đưa một nghiệm ra
```

```
Res: longint = 0;
Procedure xuli(u: longint);
Var i, p : longint;
Begin
P:=1;
For i:= 2 to u do
if not ((nguyento(a[u]) = true) and (a[u] <> a[u-1]) and nguyento(a[i-1])) then
begin
p:=0;
break;
end;
if p=1 then
begin
if u > res then
begin
res:=u;
```



```
for i:=1 to u do d[i] := a[i];
end;
end;
End;
//nhap;
Thu(1);
Writeln(res);
For i:=1 to res do write(d[i],#32)
```



```
1 #include<bits/stdc++.h>
2 #define nmax 100007
3 using namespace std;
4 int a[nmax], t[nmax], n, d[nmax], res=0;
5 bool snt(int u)
6 {
7     if(u<=1) return false;
8     if(u==2 || u==3) return true;
9     for(int i=2;i<=trunc(sqrt(u));i++)
10         if(u%i==0) return false;
11     return true;
12 }
13 void xl(int u)
14 {
15     int p=1;
16     for(int i=2;i<=u;i++)
17         if(! (snt(a[i]) && (a[i] != a[i-1]) && snt(a[i-1])))
18         {
19             p=0;
20             break;
21         }
22     if(p==1)
23     {
24         if(u>res)
25         {
26             res=u;
27             for(int i=1;i<=u;i++)
28                 d[i]=a[i];
29         }
30     }
31 }
32 void thu(int i)
33 {
34     for(int j=a[i-1];2*j<=n-t[i-1];j++)
35     {
36         a[i]=j;
37         t[i]=t[i-1]+j;
38         thu(i+1);
39     }
40     a[i]=n-t[i-1];
41     xl(i);
42 }
43 int main()
44 {
45     cin>>n;
46     a[0]=1; t[0]=0;
47     thu(1);
48     cout<<res<<"\n";
49     for(int i=1;i<=res;i++) cout<<d[i]<<" ";
50     return 0;
51 }
```



```
1 var x,t:array[-5..100007] of longint;
2 int:array[-5..100007] of boolean;
3 n,i:longint;
4
5
6 procedure prime;
7 var q,w:longint;
8 begin
9   fillchar(int,sizeof(int),true);
10  int[1]:=false;
11  q:=2;
12  while q <= trunc(sqrt(n)) do
13  begin
14    while int[q]=false do inc(q);
15    for w:=2 to n div q do int[q*w]:=false;
16    inc(q);
17  end;
18 end;
19
20
21 procedure execute(k:longint);
22 var p,l: longint;
23 begin
24   p:=1;
25   for l:=2 to k do
26     if not ((int[x[l]] = true) and (x[l] <> x[l-1])) then
27     begin
28       p:=0;
29       break;
30     end;
31   if p = 1 then
32     begin
33       for l:=1 to k do write(x[l],#32);
34       writeln;
35     end;
36 end;
37
38
39 procedure testing(i:longint);
40 var j:longint;
41 begin
42   for j:=x[i-1] to ((n-t[i-1]) div 2) do
43   begin
44     x[i]:=j;
45     t[i]:=t[i-1]+j;
46     testing(i+1);
47   end;
48   x[i]:=n-t[i-1];
49   execute(i);
50 end;
51 begin
52   readln(n); x[0]:=1; t[0]:=0;
53   prime;
54   testing(1);
55 end.
```



40) CHỌN QUÀ

Cho N món quà, món quà thứ i có giá trị $a[i]$. An, Bình, Cường và Dũng được phép chọn mỗi người một món quà có chỉ số lần lượt là i, j, k, p sao cho tổng: $a_i + a_j + a_k + a_p$ là số nguyên tố lớn nhất.

Hãy tìm cách chọn quà sao cho tổng thu được là số nguyên tố lớn nhất.

Dữ liệu: Vào từ tệp CSELECTP.INP gồm

- Dòng 1: Ghi số nguyên dương N là số lượng món quà. ($N \leq 100$).
- Dòng 2: Ghi N số nguyên dương $a[i]$. ($a[i] \leq 10^6$).

Kết quả: Ghi ra tệp CSELECTP.OUT gồm:

- Dòng 1: Tổng nguyên tố lớn nhất tìm được.
- Dòng 2: Ghi ra chỉ số của các phần tử được chọn. Nếu có nhiều phương án chọn hãy đưa ra một cách. Nếu không có phương án chọn thì ghi ra 0.

CSELECTP .INP	CSELECTP .OUT
5	13
3 1 2 5 4	1 2 4 5

Thuật toán: <https://youtu.be/odr174t1vqk>

Cách 1: Độ phức tạp $O(n^4\sqrt{4 \cdot amax})$. Chương trình chạy được với $N \leq 50$

Duyệt (i, 1, n-3)

Duyệt (j, i+1, n-2)

Duyệt(k, j+1, n-1)

Duyệt(p, k+1, n)

Nếu $\text{nguyento}(a[i]+ a[j]+a[k]+a[p]) = \text{true}$ thì

Begin

Nếu $\text{res} < a[i]+ a[j]+a[k]+a[p]$ thì

Begin

Res := $a[i]+ a[j]+a[k]+a[p];$

I0:= i; j0:=j; k0:=k; p0:=p;

End;

End;

Đưa res ra.

Đưa (i0, j0, k0, p0);

Cách 2: Độ phức tạp $O(n^4)$. <https://youtu.be/8EFt5oa-s50>

- Tìm **rmax** là giá trị lớn nhất của dãy.

- Sàng nguyên tố ($4 * rmax$); thu được mảng F[i] lưu trạng thái nguyên tố của i.

- Duyệt (i, 1, n-3)

Duyệt (j, i+1, n-2)

Duyệt(k, j+1, n-1)

Duyệt(p, k+1, n)

Nếu $F[a[i]+ a[j]+a[k]+a[p]] = \text{true}$ thì

Begin

Nếu $\text{res} < a[i]+ a[j]+a[k]+a[p]$ thì



```

Begin
    Res := a[i]+ a[j]+a[k]+a[p];
    I0:= i; j0:=j; k0:=k; p0:=p;
End;

End;

Đưa res ra.
Đưa (i0,j0,k0,p0);

```

41) QLHVHT.*

Một dãy a_1, a_2, \dots, a_N gọi là **một hoán vị hoàn toàn** của tập $\{1, 2, \dots, N\}$ nếu nó là một hoán vị và thỏa mãn $a[i] \neq i$.

Cho N . Hãy liệt kê các hoán vị hoàn toàn của tập N phân tử.

Ví dụ: $N = 3$

~~1 2 3;~~

~~1 3 2;~~

~~2 1 3;~~

~~2 3 1;~~

~~3 1 2;~~

~~3 2 1~~

Kết quả: 2 3 1; 3 1 2

QLHVHT .INP	QLHVHT .OUT
3	2 3 1 3 1 2

Thuật toán:

<https://youtu.be/JccnDYF6YhM>

- Lần lượt liệt kê các hoán vị của tập N phân tử ($1, 2, 3, \dots, N$).

- Xử lí: kiểm tra tất cả các $a[i] \neq i$

$P:=1;$

For (i, 1, N)

 Nếu $a[i] = i$ thì

 Begin

$P:= 0;$

 Break;

 End;

If $p = 1$ then

 Begin

 For(i,1,N) đưa $a[i]$ ra.

 Writeln;

 End;

42) DQHV6.*

Cho N và dãy a_1, a_2, \dots, a_N . Thực hiện đổi chỗ một số phần tử sao cho tổng.

$S = |a_1 - a_2 + a_3| + |a_2 - a_3 + a_4| + \dots + |a_{n-2} - a_{n-1} + a_n|$ đạt giá trị lớn nhất.



Dữ liệu: Vào từ tệp DQHV6.INP gồm

- Dòng 1: Ghi số nguyên dương N
- Dòng 2: Ghi N số nguyên a_i.

Kết quả: ghi ra tệp DQHV6.OUT gồm:

- Dòng 1: Ghi giá trị lớn nhất của S tìm được.
- Các dòng tiếp theo, mỗi dòng ghi một cách sắp xếp dãy đã cho để có S lớn nhất.

Thuật toán:

Sinh hoán vị vào mảng b[i].

Thủ tục xuli: cài tiến với mỗi cấu hình ta thực hiện

- Duyệt (i,1,N-2)

$$S := S + \text{abs}(a[b[i]] + a[b[i+1]] + a[b[i+2]]);$$
- Nếu res < S thì


```
Begin
  Res := S;
  For(i,1,N) d[i] := b[i];
End;
```
- Chương trình chính:
 - Đọc dữ liệu, khởi tạo mảng F[i] = 1; res := 0;
 - Thu(1);
 - Đưa res ra.
 - For(i,1,N) write(a[d[i]],#32);

43) DQHV7.*

Cho N. Thực hiện đổi chỗ các chữ số của N để được số mới. Tìm số đối xứng lớn nhất thu được.

Số đối xứng là số đọc từ trái sang phải và từ phải sang trái ta được một số.

Ví dụ: 1, 2, 3, 4....., 9, 11, 22, ...

Dữ liệu: Vào từ tệp DQHV7.INP ghi số nguyên dương N

Kết quả: ghi ra tệp DQHV7.OUT số đối xứng lớn nhất tìm được. Nếu không có số nào thì ghi ra 0.

DQHV7.INP	DQHV7.OUT
12412	21412

Thuật toán:

- Tách các chữ số của N đầy vào mảng a[i] có K phần tử.
- Sinh hoán vị vào mảng b[i].
- Với mỗi hoán vị: Tính giá trị Nmoi – hàm tinhgtN(u)
- Duyệt (i,1,N) S := S * 10 + a[b[i]];
- Nếu (res < Nmoi) và (doixung(Nmoi) = true) thì res := nmoi;
- Chương trình chính: Đưa res ra.

44) BỘI SỐ CHUNG NHỎ NHẤT

Hãy tìm **bội số chung** nhỏ nhất của **tất cả** các số nguyên từ 1 đến N,

Ví dụ: N = 3, phạm vi từ 1 tới N: 1, 2, 3. Bội chung nhỏ nhất là số nguyên đồng chia hết cho các số này.



Dữ liệu: Vào từ file văn bản LCM.INP chứa số nguyên N ($1 \leq N \leq 1\,000$).
Kết quả: Đưa ra file văn bản LCM.OUT một số nguyên - bội số chung nhỏ nhất

Ví dụ:

LCM.INP	LCM.OUT
3	6

Thuật toán:

- Video hướng dẫn: <https://youtu.be/INg8GHE2rh0>
- Công thức tìm BCNN: $BCNN(u, v) = \frac{u \cdot v}{ucln(u, v)}$

Ví dụ: $BCNN(6, 9) = 6 \cdot 9 / 3 = 18$

- Tô chức hàm tìm UCLN(u,v) – theo giải thuật O clt.
- Tô chức hàm tìm BCNN(u,v) theo công thức trên. Sử dụng phép toán div.

Function BCNN(u, v: longint): longint; Begin Exit(u*v div ucln(u,v)); End;	Int bcnn(int u, int v) { Return (u*v / __gcd(u,v)); }
-------------------------------------------------------------------------------------	----------------------------------------------------------------

- Chương trình chính:

```
Res := 1;  
For(i, 1, N)  
    Res := bcnn(res, i);
```

Đưa res ra.



f44



```

[ ]
Var res,i,n: longint;
Function Ucln(x,y: longint): longint;
Var r: longint;
Begin
    While y<>0 do
        Begin
            r:=x mod y;
            x:=y;
            y:=r;
        End;
    Ucln:=x;
End;
Function Bcnn(u,v: longint): longint;
Begin
    Exit(u*v div ucln(u,v));
End;
Begin
    Assign(Input,'LCM.INP');
    Assign(Output,'LCM.OUT');
    Reset(Input);
    Rewrite(Output);
    Res:=1;
    ReadLn(n);
    For i:=1 to n do
        Res:=Bcnn(res,i);
    Write(res);

```

```

1 #include <bits/stdc++.h>
2 using namespace std;
3 int bcnn(long long u, long long v)
4 {
5     return u*v/__gcd(u,v);
6 }
7 int main()
8 {
9     long long n; int res=1;
10    cin>>n;
11    for(int i=1;i<=n;i++)
12        res=bcnn(res,i);
13    cout<<res;
14    return 0;
15 }
16

```

45) SỐ NHỊ THẾ

Một số tự nhiên được gọi là nhị thế nếu ở cơ số 10 dạng biểu diễn của nó có không quá 2 chữ số khác nhau. Ví dụ, các số 3, 23, 100, 12121 được gọi là nhị thế, còn các số 123, 9980 thì không phải là nhị thế.

Yêu cầu: Cho số nguyên N ($1 \leq N \leq 30\,000$). Hãy tìm số nhị thế gần N nhất. Nếu tồn tại 2 số thỏa mãn thì đưa ra số lớn hơn trong 2 số đó.

Ví dụ: N = 213 xung quanh có các số nhị thế: 211, 199, 200, 221, 212...

Trong các số nhị thế này, số 212 là số gần số 213 nhất.

Dữ liệu: Vào từ file văn bản DIHYBRID.INP gồm nhiều dòng, mỗi dòng chứa một số nguyên N.



Kết quả: Đưa ra file văn bản DIHYBRID.OUT các kết quả tìm được, mỗi số trên một dòng.

Ví dụ:

DIHYBRID.IN	DIHYBRID.OUT
P	
123	122
2012	2020
11111	11111



Thuật toán:

- Video hướng dẫn: <https://youtu.be/-ES51-fO87Q>
- Hàm kiểm tra u có là số nhị phân
 - o Đếm phân phối chữ số của u.
 - o Nếu có hơn 2 chữ số xuất hiện trong u thì exit(false).
 - o Exit(true);

```
Function sonhithe(u: longint): boolean;
Var
  C: array[0..9] of longint;
  J, d: longint;
Begin
  For j:= 0 to 9 do c[j] := 0;
  While u>0 do
    Begin
      inc(C[U mod 10]);
      u:= u div 10;
    End;
  D:= 0;
  For j:=0 to 9 do
    If c[j] > 0 then
      Begin
        Inc(d);
        If d > 2 then exit(false);
      End;
  Exit(true);
End;
```

- Chương trình chính:
 - + Tìm số nhị phân bên trái của N: a:= N – 1;
Chừng nào sonhithe(a) = false làm dec(a);
 - + Tìm số nhị phân bên phải của N: b:= N + 1;
Chừng nào sonhithe(b) = false làm inc(b);



+ Nếu $|a-N| \geq |b-N|$ thì res := b

Ngược lại res := a;

+ Đưa res ra.

```
File Edit Search Run Compile Debug Tools Options Window Help
-[ ] DIHYBIRD.pas
var n,i,res,a,b:longint;
const
  fi='DIHYBIRD.inp';
  fo='DIHYBIRD.out';
function sonhithe(u:longint):boolean;
var c:array[0..9] of longint;
  j,d:longint;
begin
  for j:=0 to 9 do c[j]:=0;
  while u>0 do
    begin
      inc(c[u mod 10]);
      u:=u div 10;
    end;
  d:=0;
  for j:=0 to 9 do
    if c[j]>0 then
      begin
        inc(d);
        if d>2 then exit(false);
      end;
  exit(true);
end;
begin
  assign(input,fi); reset(input);
  assign(output,fo); rewrite(output);
  readln(n);
  a:=n-1;
  while sonhithe(a)=false do dec(a);
  b:=n+1;
  while sonhithe(b)=false do inc(b);
  if abs(a-n) >= abs(b-n) then res:=b
    else res:=a;
  write(res);
  close(input);
  close(output);
end.
37:60
#include <bits/stdc++.h>
using namespace std;
bool sonhithe(int u)
{
    int c[10];
    int j,d;
    for (int j=0;j<=9;j++) c[j]=0;
    while (u>0)
    {
        c[u%10]++;
        u=u/10;
    }
    d=0;
    for (int j=0;j<=9;j++)
        if (c[j]>0)
    {
        d++;
        if (d>2) return false;
    }
    return true;
}
int main()
{
    int n;
    int a,b,res;
    cin>>n;
    a=n-1; b=n+1;
    while (sonhithe(a)==false) a--;
    while (sonhithe(b)==false) b++;
    int t1=abs(a-n); int t2=abs(b-n);
    if (t1>=t2) res=b;
    else res=a;
    cout<<res;
    return 0;
}
```



46) THỨ TỰ TỰ ĐIỂN CÓ TRỌNG SỐ

Xét các số nguyên dương trong phạm vi từ 1 đến N ($N \leq 10^{18}$). Gọi trọng số của một số là tổng các chữ số của nó và ký hiệu trọng số của x là $w(x)$.

Người ta sắp xếp các số từ 1 đến N theo các tiêu chuẩn sắp xếp sau:

- Nếu a và b là 2 số nguyên và $w(a) < w(b)$ thì a đứng trước b ,
- Nếu $w(a) = w(b)$ thì số nào trong dạng biểu diễn ở hệ 10 có thứ tự từ điển nhỏ hơn sẽ đứng trước.

Ví dụ, theo các quy tắc trên:

- Số 120 đứng trước số 4,
- Số 555 đứng trước số 78,
- Số 20 đứng trước số 200.

Yêu cầu: Cho 2 số nguyên N và K ($1 \leq K \leq N$). Hãy xác định thứ tự của K trong cách sắp xếp trên.

Dữ liệu: Vào từ file văn bản GRLEX.INP, gồm 2 số nguyên N và K, mỗi số trên một dòng.

Kết quả: Đưa ra file văn bản GRLEX.OUT:

- Dòng thứ nhất chứa thứ tự của số K,
- Dòng thứ 2 chứa số nguyên có số thứ tự là K.

Ví dụ:

GRLEX.INP	GRLEX.OUT
20	2
10	14



Thuật toán:

- Video hướng dẫn: <https://youtu.be/EGXXGEYjvhc>
- Tổ chức hàm tongcs(u) – tính tổng chữ số của u.
- Tổ chức thủ tục sắp xếp với hàm điều kiện:
 - Nếu a và b là 2 số nguyên và $w(a) < w(b)$ thì a đứng trước b ,
 - Nếu $w(a) = w(b)$ thì số nào trong dạng biểu diễn ở hệ 10 có thứ tự từ điển nhỏ hơn sẽ đứng trước

tongcs(u)
Function cmp(u, v: longint): boolean;
Var



Begin

```
P:= tongcs(u); q:= tongcs(v);
If p < q then exit(false);
If p = q then
  If u > v then exit(true) else exit(false);
Exit(true);
End;
```

- Tạo mảng a chứa các số nguyên từ 1 tới N:

```
For(i,1,N) a[i] := i;
```

- For(i, 1, N - 1)

```
For (j, i+1, N)
```

```
  If cmp(a[i],a[j]) then đổi chỗ(a[i],a[j]);
```

- For(i,1, N) nếu a[i] = k then

```
  begin writeln(i); break; end;
```

- Đưa a[k] ra.

```
Free Pascal IDE
File Edit Search Run Compile Debug Tools Options Window
GRLEX.pas
[1]
var n,k,i,j,tg:longint;
  a:array[-..10007] of longint;
const
  fi='GRLEX.inp';
  fo='GRLEX.out';
function tongcs(u:longint):longint;
var s:longint;
begin
  s:=0;
  while u>0 do
    begin
      s:=s+u mod 10;
      u:=u div 10;
    end;
  exit(s);
end;
function cmp(u,v:longint):boolean;
var p,q:longint;
begin
  p:=tongcs(u); q:=tongcs(v);
  if p<q then exit(false);
  if p=q then
    if u>v then exit(true) else(exit(false));
  exit(true);
end;
begin
  assign(input,fi); reset(input);
  assign(output,fo); rewrite(output);
  readln(n,k);
  for i:=1 to n do a[i]:=i;
  for i:=1 to n-1 do
    for j:=i+1 to n do
      if cmp(a[i],a[j]) then
        begin
          tg:=a[i];
          a[i]:=a[j];
          a[j]:=tg;
        end;
  for i:=1 to n do write(a[i],#32);
  writeln(a[k]);
  close(input);
  close(output);
end.
```



```
1 #include <bits/stdc++.h>
2 #define nmax 100007
3 using namespace std;
4 int a[nmax];
5 int tongcs(int u)
6 {
7     int tach,s=0;
8     while(u!=0)
9     {
10         tach=u%10;
11         u=u/10;
12         s+=tach;
13     }
14     return s;
15 }
16 bool cmp(int u, int v)
17 {
18     int p=tongcs(u); int q=tongcs(v);
19     if(p<q) return false;
20     if(p==q)
21         if(u>v) return true;
22     else return false;
23     return true;
24 }
25 int main()
26 {
27     int n,k;
28     cin>>n>>k;
29     for(int i=1;i<=n;i++)
30         a[i]=i;
31     for(int i=1;i<=n-1;i++)
32         for(int j=i+1;j<=n;j++)
33             if(cmp(a[i],a[j])) swap(a[i],a[j]);
34     for(int i=1;i<=n;i++)
35         if(a[i]==k)
36         {
37             cout<<i<<" ";
38             break;
39         }
40     cout<<a[k];
41     return 0;
42 }
```

47) ĐÉM CHỮ SỐ

Cho 2 số nguyên a và b ($0 < a, b \leq 100\,000\,000$). Xét tất cả các số nguyên nằm giữa a và b (kể cả 2 số này).

Hãy **đếm** số lượng từng chữ số thập phân xuất hiện trong tất cả các số đó.

Ví dụ: $a = 1$; $b = 12$;

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12



Ví dụ, với $a = 1024$ và $b = 1032$, ta có dãy số nguyên :

1024 1025 1026 1027 1028 1029 1030 1031 1032

Trong dãy này có 10 số 0, 10 số 1, 7 số 2, 3 số 3 ...

Dữ liệu: Vào từ file văn bản COUNTING.INP gồm nhiều dòng (nhưng không quá 500), mỗi dòng ứng với một test, chứa 2 số nguyên a và b .

Kết quả: Đưa ra file văn bản COUNTING.OUT, kết quả mỗi test trên một dòng, gồm 10 số nguyên cho biết số lượng các chữ số từ 0 đến 9.

Ví dụ:

COUNTING.INP	COUNTING.OUT
1 10	1 2 1 1 1 1 1 1 1 1
44 497	85 185 185 185 190 96 96 96 95 93
346 542	40 40 40 93 136 82 40 40 40 40
1199 1748	115 666 215 215 214 205 205 154 105 106
1496 1403	16 113 19 20 114 20 20 19 19 16
1004 503	107 105 100 101 101 197 200 200 200 200
1714 190	413 1133 503 503 503 502 502 417 402 412
1317 854	196 512 186 104 87 93 97 97 142 196
1976 494	398 1375 398 398 405 499 499 495 488 471
1001 1960	294 1256 296 296 296 296 287 286 286 247

Thuật toán:

- Video hướng dẫn: <https://youtu.be/hTbuXiUzVtw>
- Tổ chức thủ tục đếm phân phối các chữ số của u vào mảng C.
- Kết quả của bài toán lưu số lần xuất hiện (i, c[i] lần).

```
Procedure dempp(u: longint);
```

```
Begin
```

```
    While u > 0 do
```

```
        Begin
```

```
            Inc(c[u mod 10]);
```

```
            U:= u div 10;
```

```
        End;
```

```
    End;
```

- Chương trình chính: // c: array[0..9] of longint;
 - o For(i,0,9) c[i] := 0;
 - o Duyệt (i, a, b)
 - Dempp(i);
 - o For i:= 0 to 9 do write(c[i],#32);





```
1 #include <bits/stdc++.h>
2 #define nmax 100007
3 using namespace std;
4 int c[nmax];
5 void dempp(int u)
6 {
7     while (u>0)
8     {
9         c[u%10]++;
10        u=u/10;
11    }
12 }
13 int main()
14 {
15     int a,b;
16     cin>>a>>b;
17     for (int i=0;i<=9;i++) c[i]=0;
18     for (int i=a;i<=b;i++) dempp(i);
19     for (int i=0;i<=9;i++) cout<<c[i]<<" ";
20
21     return 0;
22 }
```

Free Pascal

```
File Edit Search Run Compile Debug Tools Options Window Help
[ ] bai4_counting.pas
uses math;
const
  fi = 'counting.inp';
  fo = 'counting.out';
var a,b,i : longint;
  c : array[0..9] of longint;
  procedure dempp(u : longint);
begin
  while u > 0 do
  begin
    inc(c[u mod 10]);
    u := u div 10;
  end;
end;
begin
  assign(input,fi); reset(input);
  assign(output,fo); rewrite(output);
  readln(a,b);
  for i := 0 to 9 do c[i] := 0;
  for i := a to b do
    dempp(i);
  for i := 0 to 9 do
    write(c[i],#32);
  close(input); close(output);
end.
```

4:17

F1 Help F2 Save F3 Open Alt+F9 Compile F9 Make Alt+F10 Local me

48) SỐ ĐẸP

Liza coi một số ở hệ 10 là đẹp nếu nó không chứa các chữ số nào khác ngoài số chữ 0 và k ($1 \leq k \leq 9$). Ví dụ, với $k = 2$, các số đẹp là 2, 20, 2002, ... Liza không cảm tình với các số khác, vì vậy bạn ấy luôn tìm cách biểu diễn chúng **thành tổng** các số đẹp. Ví dụ, nếu $k = 3$ thì 69 có thể biểu diễn dưới dạng $69 = 33+30+3+3$.

Yêu cầu: Cho n và số k ($1 \leq n \leq 10^9$).

Hãy xác định các cách biểu diễn n thành tổng các số đẹp.



Dữ liệu: Vào từ file văn bản NUMBERS.INP gồm nhiều dòng, mỗi dòng chứa một số tự nhiên n và chữ số k .

Kết quả: Đưa ra file văn bản NUMBERS.OUT: Gồm nhiều dòng, mỗi dòng ghi một cách phân tích n . Nếu không có cách phân tích nào thì ghi ra -1.

Ví dụ:

NUMBERS.INP	NUMBERS.OUT
69 3	33+33+3
6 5	33+30+3+3
10 9	-1
	-1



h23

Thuật toán:

Video hướng dẫn: <https://youtu.be/L6V26iyxPuY>

Tổ chức hàm kiểm tra sodep(u):

- Đếm phân phối các chữ số của u vào mảng $C[i]$.
- Nếu $c[0] = 0$ và $c[k] = 0$ thì exit(fase);
- Các số từ 0 tới 9 (trừ số 0 và số K bắt buộc xuất hiện) mà xuất hiện thì exit(false);
- Exit(true);

```
Function sodep(u: longint): boolean;
Var c: array[0..9] of longint;
Begin
  For(i, 0, 9) c[i] := 0;
  While u > 0 do
    Begin
      Inc(c[u mod 10]);
      U:= u div 10;
    End;
  If (c[0] = 0) và (c[k] =0) then exit(false);
  For(i,0,9)
    Nếu (c[i] > 0) và (i <> 0) và (i <> k) exit(false);
  Exit(true);
End;
```

```
Procedure xuli(u: longint);
Var p: longint;
Begin
  P:=1;
  For (i,1, u)
    Nếu sodep(a[i]) = false thì
      Begin
```



```
P:=0;  
Break;  
End;  
Nếu p = 1 thì  
Begin  
    For(i,1,u) đưa a[i] ra;  
    xuống dòng;  
    end;  
End;
```

```
Đệ quy phân tích số  
Procedure phantich(i: longint);  
Var  
Begin  
    For j:= a[i-1] to (N – T[i-1]) div 2 do  
        Begin  
            A[i] := j;  
            T[i]:= t[i-1] + j;  
            Phantich(i+1);  
        End;  
        A[i]:= N – T[i-1];  
        Xuli(i);  
    End;  
Chương trình chính:  
A[0]:=1; T[0]:= 0;  
Phantich(1);
```



```
1 #include <bits/stdc++.h>
2 #define maxx 100007
3 using namespace std;
4 int a[maxx], n, k, t[maxx], q;
5 bool sodep(int u)
6 {
7     int c[10];
8     for(int i=0;i<=9;i++) c[i]=0;
9     while(u>0)
10    {
11        c[u%10]++;
12        u=u/10;
13    }
14    if(c[0]==0&&c[k]==0) return false;
15    for(int i=0;i<=9;i++)
16        if(c[i]>0&&i!=0&&i!=k) return false;
17    return true;
18 }
19 void xuli(int u)
20 {
21     int p=1;
22     for(int i=1;i<=u;i++)
23         if(sodep(a[i])==false)
24         {
25             p=0;
26             break;
27         }
28     if(p==1)
29     {
30         q=1;
31         for(int i=1;i<=u;i++) cout<<a[i]<<" ";
32         cout<<endl;
33     }
34 }
35 void thu(int i)
36 {
37     for(int j=a[i-1];j<=(n-t[i-1])/2;j++)
38     {
39         a[i]=j;
40         t[i]=t[i-1]+j;
41         thu(i+1);
42     }
43     a[i]=n-t[i-1];
44     xuli(i);
45 }
46 int main()
47 {
48     cin>>n>>k;
49     q=0; a[0]=k; t[0]=0;
50     thu(1);
51     if(q==0) cout<<-1;
52     return 0;
53 }
```



Free Pascal IDE

File Edit Search Run Compile Debug Tools Options Window Help

[] NUMBERS.pas 4=[↑]=

```
var N,K,Q:longint;
    A,T:array [-7..100007] of longint;
function SODEP(u:longint):boolean;
var i:longint;
    C:array [0..9] of longint;
begin
    for i:=0 to 9 do
        C[i]:=0;
    while u > 0 do
    begin
        inc(C[u mod 10]);
        u:=u div 10;
    end;
    if (C[0] = 0) and (C[k] = 0) then exit(false);
    for i:=0 to 9 do
        if (C[i] > 0) and (i <> 0) and (i <> k) then exit(false);
    exit(true);
end;
procedure xuli(u:longint);
var P,i:longint;
begin
    10:51
F1 Help F2 Save F3 Open Alt+F9 Compile F9 Make Alt+F10 Local menu
```



Free Pascal IDE

File Edit Search Run Compile Debug Tools Options Window Help

[] NUMBERS.pas 4=[↑]=

```
begin
    P:=1;
    for i:=1 to u do
        if SODEP(A[i]) = false then
            begin
                P:=0;
                break;
            end;
    if P = 1 then
        begin
            Q:=1;
            for i:=1 to u do
                write(A[i],#32);
                writeln;
            end;
        end;
procedure Phantich(i:longint);
var
    j:longint;
begin
    for j:=A[i-1] to (N - T[i-1]) div 2 do
        begin
            if SODEP(j) = true then
                begin
                    A[i]:=j;
                    Phantich(i+1);
                end;
        end;
end;
F1 Help F2 Save F3 Open Alt+F9 Compile F9 Make Alt+F10 Local menu
```



Free Pascal IDE

File Edit Search Run Compile Debug Tools Options Window Help

[] NUMBERS.pas 4=[↑]=

```
for j:=A[i-1] to (N - T[i-1]) div 2 do
begin
    A[i]:=j;
    T[i]:=T[i-1] + j;
    Phantich(i+1);
end;
A[i]:=N - T[i-1];
xuli(i);
end;
begin
    assign(input, 'NUMBERS.inp');reset(input);
    assign(output, 'NUMBERS.out');rewrite(output);
    readln(N,K);
    Q:=0;
    A[0]:=K;
    T[0]:=0;
    Phantich(1);
    if Q = 0 then write(-1);
    close(input);
    close(output);
end.
```

60:27

F1 Help F2 Save F3 Open Alt+F9 Compile F9 Make Alt+F10 Local menu

49) UỐC SÓ CHUNG LỚN NHẤT

Xét tam giác Pascal:

0		1		
1		1	1	
2		1	2	1



3 1 3 3 1
4 1 4 6 4 1

.....



Yêu cầu: Hãy tìm ước số chung lớn nhất của các số lớn hơn 1 trong dòng thứ N của tam giác Pascal ($1 < N < 2^{31}$).

Dữ liệu: Vào từ file văn bản COMDIV.INP gồm nhiều dòng, mỗi dòng chứa một số nguyên N.

Kết quả: Đưa ra file văn bản COMDIV.OUT, mỗi kết quả (nguyên) đưa ra trên một dòng.

Ví dụ:

COMDIV.INP
3
4

COMDIV.OUT
3
2



Thuật toán:

- Video hướng dẫn: <https://youtu.be/ouRzW3GLwPo>
- Hàm ước ucln(u,v) - Óclit
 - Dùng mảng một chiều gồm N phần tử, khởi tạo 0.
 - A[1] = 1;
 - For(i,1,n)
 - Begin
 - Fordownto(j,i+1, 2) a[j]:= a[j] + a[j-1];
 - // For(j,1, i+1) write(a[j],#32);
 - // Writeln();
 - End;
 - (1) Tìm phần tử đầu tiên khác 1 lưu vào res.
 - For(i,1,n) nếu a[i] <> 1 thì
 - begin
 - res := a[i]; i0:= i;
 - break;
 - end;
 - For(i,i0,n)
 - Nếu a[i] <> 1 thì res := ucln(res,a[i]);
 - Đưa res ra.



```
1 #include <bits/stdc++.h>
2 #define nmax 100007
3 using namespace std;
4 int n,a[nmax],res,i0;
5 int main()
6 {
7     cin>>n;
8     a[1]=1;
9     for(int i=1;i<=n;i++)
10    {
11        for(int j=i+1;j>=2;j--) a[j]=a[j]+a[j-1];
12    }
13    for(int i=1;i<=n;i++)
14    {
15        if(a[i]!=1)
16        {
17            res=a[i]; i0=i;
18            break;
19        }
20    }
21    for(int i=i0;i<=n;i++)
22    {
23        if(a[i]!=1) res=__gcd(res,a[i]);
24    }
25    cout<<res;
26 }
```



Free Pascal IDE

```
File Edit Search Run Compile Debug Tools Options Window
[comdiv.pas] comdiv = comdiv.pas
var
  n, res, i, j, i0: longint;
  a: array[-7..100007] of longint;
function ucln(u, v: longint): longint;
var r: longint;
begin
  while v > 0 do
    begin
      r := u mod v;
      u := v;
      v := r;
    end;
  exit(u);
end;

begin
  assign(input,'comdiv.inp');reset(input);
  assign(output,'comdiv.out');rewrite(output);
  readln(n);
  a[1] := 1;
  for i := 1 to n do
    for j := i + 1 downto 2 do a[j] := a[j] + a[j-1];
  for i := 1 to n do
    if a[i] <> 1 then
      begin
        res := a[i];i0 := i;
        for j := i + 1 downto 2 do a[j] := a[j] + a[j-1];
        for i := 1 to n do
          if a[i] <> 1 then
            begin
              res := a[i];i0 := i;
              break;
            end;
        for i := i0 to n do
          if a[i] <> 1 then res := ucln(res, a[i]);
        writeln(res);
        close(input);close(output);
      end;
end.
```



Giả thiết A là một xâu. Xâu con của A là xâu nhận được từ A bằng cách gạch bỏ một số ký tự (không nhất thiết phải liên tiếp nhau) trong A.

Yêu cầu: Cho 2 xâu A và B chỉ chứa các chữ cái la tinh thường, độ dài mỗi xâu không quá 2 000. Hãy xác định xâu con ngắn nhất của A **không phải là xâu con** của B.

Dữ liệu: Vào từ file văn bản SUBSTR.INP gồm 2 dòng, dòng thứ nhất chứa xâu A, dòng thứ 2 chứa xâu B.

Kết quả: Đưa ra file văn bản SUBSTR.OUT:

Dòng đầu tiên chứa số nguyên N - độ dài của xâu con ngắn nhất tìm được,
Dòng thứ 2: xâu tìm được.

Ví dụ:

SUBSTR.INP	SUBSTR.OUT
banana anbnaanbaan	2 nn



Thuật toán:

- **Video hướng dẫn:** <https://youtu.be/qZnM4dD-h8A>
- Sinh dãy nhị phân độ dài N = length(A).
- Với mỗi cấu hình thu được, nối các phần tử được chọn của xâu A vào xâu kq.
Tam:=""";
For(i,1,n)
 Nếu c[i] = 1 thì tam:= tam + a[i]; //kq, a, b: string;
 Nếu length(tam)> 0 thì
 Nếu pos(tam,b) = 0 thì
 Nếu length(tam) < res thì
 Begin
 Res := length(tam);
 Kq := tam;
 End;
 End;
- Chương trình chính:
Nhập a, b;
Res := length(a); n:= length(a);
Thu(1);
Đưa res ra;
Đưa kq ra.



Free Pascal IDE



```
File Edit Search Run Compile Debug Tools Options Window substr
[ ]= substr
var
  a, b, kq: string;
  i, n, res: longint;
  d: array[-7..1000007] of longint;
procedure xuli();
var tam: string = '';
  p: longint;
begin
  for p := 1 to n do
    if d[p] = 1 then tam := tam + a[p];
  if length(tam) > 0 then
    if pos(tam, b) = 0 then
      if length(tam) < res then
        begin
          res := length(tam);
          kq := tam;
        end;
  end;
procedure thu(i: longint);
var j: longint;
begin
  for j := 0 to 1 do
    begin
      d[i] := j;
      if i = n then xuli
      else thu(i+1);
    end;
end;
begin
  assign(input,'substr.inp');reset(input);
  assign(output,'substr.out');rewrite(output);
  readln(a);
  readln(b);
  res := length(a);n := length(a);
  thu(1);
  writeln(res);
  writeln(kq);
  close(input);close(output);
end.
```



```
1 #include <bits/stdc++.h>
2 #include <string.h>
3 #define nmax 100007
4 using namespace std;
5 int n,c[nmax],res;
6 string kq;
7 string a,b;
8 void xl()
9 {
10     string tam="";
11     for(int i=0;i<=n-1;i++)
12         if(c[i]==0) tam=tam+a[i];
13     if(tam.size()>0&&b.find(tam)>b.size()&&tam.size()<res)
14     {
15         res=tam.size();
16         kq=tam;
17     }
18 }
19 void thu(int i)
20 {
21     for(int j=0;j<=1;j++)
22     {
23         c[i]=j;
24         if(i==n-1) xl();
25         else thu(i+1);
26     }
27 }
28 int main()
29 {
30     freopen("SUBSTR.inp","r",stdin);
31     freopen("SUBSTR.out","w",stdout);
32     getline(cin,a);
33     getline(cin,b);
34     res=a.size(); n=a.size();
35     thu(0);
36     cout<<res<<endl;
37     cout<<kq;
38     return 0;
39 }
```



51) PALINDROME



Palindrome là xâu mà ta có thể đọc từ trái sang phải hoặc từ phải sang trái đều như nhau. Ví dụ, các các xâu Z, TOT và MADAM là những palindrome, còn xâu ADAM – không phải là palindrome.

Từ một xâu bất kỳ ta có thể gạch bớt một số ký tự, để các ký tự còn lại tạo thành một xâu palindrome. Hai lần gạch được coi là khác nhau, nếu ít nhất có một vị trí ký tự bị gạch khác nhau. Số ký tự gạch có thể là 0.

Ví dụ, với xâu ABA, ta có thể có 5 cách gạch: hoặc không gạch ký tự nào hoặc gạch các ký tự gạch chân: ABA, ABA, ABA, ABA.

Yêu cầu: Cho xâu các chữ cái Latin in hoa độ dài L ($0 < L \leq 60$). Hãy cho biết số cách gạch khác nhau để nhận được xâu palindrome.

Dữ liệu: Vào từ file văn bản PALIN.INP:

Dòng đầu tiên chứa số nguyên T - số lượng Tests, ($1 \leq T \leq 15$),
T dòng sau: mỗi dòng chứa một xâu ký tự chữ cái La tinh in hoa.

Kết quả: Đưa ra file văn bản PALIN.OUT T số nguyên, mỗi số trên một dòng - số cách gạch khác nhau đối với mỗi xâu.

Ví dụ:

PALIN.INP	PALIN.OUT
3	22
BAOBAB	15
AAAA	5
ABA	

Thuật toán:

- **Video hướng dẫn:** https://youtu.be/3LCp_WaSSWU
- Tô chức hàm kiểm tra xâu u có là xâu đối xứng – palindrome:

```
bool palin(string u);
```
- Tô chức thủ tục xuli();

```
For(i,0, length(s)-1)
    If c[i] = 1 then tam := tam + s[i];
```

Nếu palin(tam) = true thì inc(res);

- Thủ tục thu(int i); // Sinh dãy nhị phân độ dài N = Length(S); điền vào vị trí [0, N-1] của mảng C.
- Chương trình chính:
 - o N := length(S);
 - o Res := 0;
 - o Thu(0);
 - o Đưa res.



```
1 #include <bits/stdc++.h>
2 using namespace std;
3 string s; int res; string n;
4 int c[100007];
5 bool palin(string u)
6 {
7     string tam="";
8     for (int i=u.length()-1; i>=0; i--)
9         tam=tam+u[i];
10    if (tam==u) return true;
11    else return false;
12 }
13 void xl()
14 {
15     string tam="";
16     for(int i=0;i<=s.length()-1;i++)
17         if(c[i]==1) tam=tam+s[i];
18     if(palin(tam)&&tam!="") res++;
19 }
20 void thu(int i)
21 {
22     for(int j=0;j<=1;j++)
23     {
24         c[i]=j;
25         if(i==s.length()-1) xl();
26         else thu(i+1);
27     }
28 }
29 int main()
30 {
31     getline(cin,s);
32     res=0;
33     thu(0);
34     cout<<res;
35     return 0;
36 }
37 }
```

52) THU GỌN

Cho số nguyên N. Người ta tạo ra số nguyên N1 bằng cách viết liên tiếp nhau các số nguyên từ 1 đến N. Ví dụ, với N = 11, ta có N1 = 1234567891011. Người ta thu gọn N1 bằng cách lần lượt xoá tất cả các chữ số ở vị trí chẵn, sau đó xoá tất cả các chữ số ở vị trí lẻ, rồi lại xoá các chữ số ở vị trí chẵn, . . . cho đến khi chỉ còn lại một chữ số:

1234567891011 → 1357901 → 370 → 30 → 0

Yêu cầu: Cho số nguyên N ($1 \leq N \leq 99\ 999$). Hãy xác định chữ số nhận được sau quá trình thu gọn số N1 tương ứng.

Dữ liệu: Vào từ file văn bản SHORTEN.INP gồm nhiều dòng, mỗi dòng chứa một số nguyên N.



Kết quả: Đưa ra file văn bản SHORTEN.OUT các chữ số còn lại ứng với các dữ liệu vào, mỗi chữ số đưa ra trên một dòng.

Ví dụ:

SHORTEN.INP	SHORTEN.OUT
1	1
4	3



Thuật toán:

- Video hướng dẫn: <https://youtu.be/orA62SNXYgk>
- Tạo xâu S gồm các số nguyên từ 1 tới N bằng cách: s:="";
For(i,1,N)


```

    {
      Đổi số i thành xâu tam; // tam =to_string(i);
      S := S + tam;
    }
  
```
- Thực hiện xoá theo đề bài:
d:= 0;
Chừng nào length(S) > 1 làm
+ nếu d mod 2 = 0 thì


```

    {
      tam = "";
      for(i,0, length(s)-1)
        if i mod 2 = 0 then tam:= tam + s[i]; //chọn các số ở vị trí lẻ
        s:= tam;
        inc(d);
    }
  
```
- Ngược lại


```

    {
      tam = "";
      for(i,0, length(s)-1)
        if i mod 2 = 1 then tam:= tam + s[i]; //chọn các số ở vị trí chẵn
        s:= tam;
        inc(d);
    }
  
```
- Đưa S ra.

```

1 #include <bits/stdc++.h>
2 using namespace std;
3 string s; int res; string n;
4 int c[100007];
5 int main()
6 {
7     int n;
8     string s="", tam;
9     cin>>n;
10    res=0;
11    for(int i=1;i<=n;i++)
12    {
13        tam=to_string(i);
14        s=s+tam;
15    }
16    int d=0;
17    while(s.length()>1)
18    {
19        if(d%2==0)
20        {
21            tam="";
22            for(int i=0;i<=s.length()-1;i++)
23                if(i%2==0) tam=tam+s[i];
24            s=tam;
25            d++;
26        }
27        else
28        {
29            tam="";
30            for(int i=0;i<=s.length()-1;i++)
31                if(i%2==1) tam=tam+s[i];
32            s=tam;
33            d++;
34        }
35        cout<<s;
36    }
37 }
38

```

53) TAM GIÁC PASCAL

Tam giác PASCAL là tam giác vô hạn có dạng:

			1									
			1	1	2	1						
			1	3	3	1						
			1	4	6	4	1					
			1	5	10	10	5	1				
1	6	15	20	15	6	1						
1	7	21	35	35	21	7	1					
...

Các dòng của tam giác được đánh số từ trên xuống dưới bắt đầu từ 0. Ở *mỗi dòng*, các cột cũng được đánh số bắt đầu từ 0. Phần tử đầu tiên ở mỗi cột là 1. Dòng thứ i chứa $i+1$ số. Gọi $a_{i,j}$ là phần tử thứ j ở dòng i . Ta có: $a_{i,j} = a_{i-1,j-1} + a_{i-1,j}$. Công thức này cũng đúng với các phần tử ở cuối dòng nếu ta cho $a_{i,j} = 0$ với $\forall j > i$.



Yêu cầu: Cho số nguyên N ($0 \leq N \leq 2*10^9$). Hãy xác định trong dòng thứ N của tam giác PASCAL có bao nhiêu số lẻ.



Dữ liệu: Vào từ file văn bản PS.INP gồm nhiều dòng, mỗi dòng chứa một số nguyên N.

Kết quả: Đưa ra file văn bản PS.OUT các số nguyên là kết quả tìm được ứng với mỗi N trong input, mỗi số trên một dòng.

Ví dụ:

PS.INP	PS.OUT
0	1
5	4
7	8



Thuật toán:

- **Video hướng dẫn:** <https://youtu.be/3t2A8pwtxGM>
- Tìm hàng thứ N của tam giác Pascal.
- For(i,1,N)
 - For(j,i+1, 2) $a[j] = a[j] + a[j-1];$
- Đếm số lượng số lẻ trên hàng N.
- For(i,1,N)
 - Nếu $a[i] \bmod 2 = 1$ thì res++;
- Đưa res ra.
- Lưu ý: Đánh chỉ số hàng bắt đầu từ hàng 0.

```

1 #include <bits/stdc++.h>
2 #define nmax 100007
3 using namespace std;
4 int n,a[nmax],res,i0;
5 int main()
6 {
7     cin>>n;
8     res=0;
9     a[0]=1;
10    for(int i=0;i<=n-1;i++)
11    {
12        for(int j=i+1;j>=1;j--) a[j]=a[j]+a[j-1];
13        for(int j=0;j<=i+1;j++)
14            cout<<a[j]<<" ";
15        cout<<endl;
16    }
17
18    for(int i=0;i<=n+1;i++)
19        if(a[i]%2==1) res++;
20    cout<<res;
21    return 0;
22
23 }
```



54) TỔNG

Từ một số có 3 chữ số cho trước người ta xây dựng chuỗi vô hạn các chữ số theo cách sau: lấy tổng 3 chữ số cuối cùng của chuỗi và điền tổng nhận được vào cuối chuỗi. Ví dụ, với số ban đầu là 123 ta có chuỗi vô hạn bắt đầu là 12361181091010, còn với số 971 ta có 971179171715... Dãy số tìm được có thể coi như là phần thập phân của một số có phần nguyên bằng 0.

Yêu cầu: Xác định chữ số thứ N trong phần thập phân của số A cho trước.

Dữ liệu: Vào từ file văn bản SUM.INP:

- Dòng thứ nhất chứa một số nguyên A có 3 chữ số,
- Các dòng tiếp theo: mỗi dòng chứa một số nguyên N ($1 \leq N < 10^{100}$).

Kết quả: Đưa ra file văn bản SUM.OUT các chữ số tìm được, mỗi số trên một dòng.

Ví dụ:

SUM.INP	SUM.OUT
123	6
4	1
8	

Thuật toán:

- **Video hướng dẫn:** <https://youtu.be/uCtecNsTfDA>

- Đọc dữ liệu vào xâu A.

- $S = A;$

- Xây dựng xâu S đến khi có độ dài lớn hơn N:

Chừng nào $S.size() \leq N$ do

- Tính tổng 3 chữ số cuối của xâu S.
 - $\text{Int Tam} = (S[S.length()-1] - '0') + (S[S.length()-2] - '0') + (S[S.length()-3] - '0');$
- Đổi tổng thành xâu Stong: $stong = to_string(tam);$
- Nối Stong vào cuối xâu S: $S = S + stong;$

- Đưa kí tự thứ N ra: Đưa $S[N-1]$. // Vì xâu trong C++ bắt đầu từ 0.

```
1 #include <bits/stdc++.h>
2 #define nmax 100007
3 using namespace std;
4 int n;
5 string a;
6 string stong;
7 int main()
8 {
9     getline(cin,a);
10    cin>>n;
11    string s=a;
12    while(s.length()<=n)
13    {
14        int tam=(s[s.length()-1]-‘0’)+(s[s.length()-2]-‘0’)+(s[s.length()-3]-‘0’);
15        //cout<<tam<<endl;
16        stong=to_string(tam);
17        s=s+stong;
18    }
19    cout<<s[n-1];
20    return 0;
21 }
22 }
```



55) TỔNG HAI SỐ

Có N lá bài, trên mỗi lá bài có ghi một số nguyên $a[i]$. Người ta muốn trải lá bài thành một dãy, sao cho tổng cực đại của các cặp số cạnh nhau là nhỏ nhất.

Ví dụ: Với dãy số 2 3 9 17; 2 3 17 9; 2 9 3 17; 2 9 17 3; 2 17 3 9; 2 17 9 3; 17 2 3 9 ...

(26; 26; 20; 26; 20; 26; 19 ...) có tổng bé nhất là 19 tương ứng với cấu hình 17 2 3 9.

Dữ liệu: Vào từ file văn bản SUMTWO.INP:

- Dòng đầu tiên chứa số nguyên N ($2 \leq N \leq 239017$),
- Dòng thứ 2 chứa N số ghi trên các quân bài.

Kết quả: Đưa ra file văn bản SUMTWO.OUT:

- Dòng đầu tiên chứa tổng lớn nhất của 2 số cạnh nhau,
- Dòng thứ 2 chứa N số xác định một cấu hình sắp xếp.

Ví dụ:

SUMTWO.INP
4
2 3 9 17

SUMTWO.OUT
19
17 2 3 9



Thuật toán:

- **Video hướng dẫn:** <https://youtu.be/pMRVN3LINBg>
- Tổ chức thủ tục Liệt kê các hoán vị của dãy có N phần tử.
- Cải tiến thủ tục xử lí:
 - Tìm tổng hai số liên tiếp lớn nhất lưu R_{max} .
 - Nếu $res > R_{max}$ thì
 - $Res = R_{max};$
 - Lưu cấu hình từ b sang mảng c.
- Chương trình chính:

```
For(i,1,N) fre[i] = true; res = trunc(1e9);
Thu(1);
Đưa res ra;
For(i,1,N) đưa a[c[i]].
```



```
Void xuli()
{
    Rmax = -trunc(1e9);
    For(p,1,n-1)
        Rmax = max(rmax, a[b[p]] + a[b[p+1]]);
    If (rmax > res)
    {
        Res = Rmax;
        For(p,1,n) c[p] = b[p];
    }
}

Void thu(int i); //sinh hoán vị
{
    For (j,1,n)
        If (fre[j])
        {
            B[i] := j;
            If (i == N) xuli();
            Else
            {
                Fre[j] = false;
                Thu(i+1);
                Fre[j] = true;
            }
        }
}
```

```

1 #include <bits/stdc++.h>
2 #define nmax 100007
3 using namespace std;
4 int n,a[nmax],b[nmax],c[nmax],rmax,res;
5 bool fre[nmax];
6 void xl()
7 {
8     rmax=trunc(1e9);
9
10    for(int p=1;p<=n-1;p++)
11        rmax=max(rmax,a[b[p]]+a[b[p+1]]);
12    if(rmax<res)
13    {
14        res=rmax;
15        for(int p=1;p<=n;p++) c[p]=b[p];
16    }
17}
18 void thu(int i)
19 {
20    for(int j=1;j<=n;j++)
21        if(fre[j])
22        {
23            b[i]=j;
24            if(i==n) xl();
25            else
26            {
27                fre[j]=false;
28                thu(i+1);
29                fre[j]=true;
30            }
31        }
32    }
33 int main()
34 {
35     freopen("SUMTWO.inp","r",stdin);
36     freopen("SUMTWO.out","w",stdout);
37     cin>>n;
38     for(int i=1;i<=n;i++) cin>>a[i];
39     for(int i=1;i<=n;i++) fre[i]=true;
40     res=trunc(1e9);
41     thu(1);
42     cout<<res<<endl;
43     for(int i=1;i<=n;i++) cout<<a[c[i]]<<" ";
44     return 0;
45 }

```

56) GIÁ TRỊ GẦN NHẤT

Cho dãy N số nguyên. Hãy đặt vào giữa những số này các dấu phép tính + và - để nhận được một biểu thức cho giá trị gần nhất có thể được với số A cho trước.

Dữ liệu: Vào từ file văn bản NEAREST.INP:

- Dòng đầu tiên chứa 2 số nguyên N A ($1 \leq N \leq 10\ 000$, $|A| \leq 10\ 000$),



- Dòng thứ i trong N dòng tiếp theo chứa số nguyên X_i , $|X_i| \leq 10\,000$,
Tổng giá trị tuyệt đối các số trong dãy cũng không vượt quá 10 000.

Kết quả: Đưa ra file văn bản NEAREST.OUT:

- Dòng đầu tiên chứa số nguyên – giá trị của biểu thức,
- Dòng thứ 2: chứa bản thân biểu thức.

Chú ý: không cần đặt ngoặc trước số âm.

Ví dụ:

NEAREST.INP	NEAREST.OUT
3 0	0
3	3+-2-1
-2	
1	



57) ĐÉM SỐ LUỢNG

Cho 3 số nguyên không âm N, A và B. Hãy tính số lượng số nguyên trong khoảng [A, B] mỗi số chứa N như một xâu con.

Ví dụ, với $N = 3$, $A = 0$ và $B = 17$ ta có 2 số (các số 3 và 13).

0, 1, 2, **3**, 4, 5, 6, 7, 8, 9, 10, 11, 12, **13**, 14, 15, 16, 17

Dữ liệu: Vào từ file văn bản COUNT.INP, gồm nhiều Tests, mỗi dòng chứa 3 số nguyên A B N ứng với một bộ dữ liệu cần tính và kết thúc bằng dòng chứa 3 số -1 -1 -1, dòng này không cần tính.
($0 \leq N \leq 999$), ($0 \leq A \leq B \leq 2\,000\,000\,000$)

Kết quả: Đưa ra file văn bản COUNT.OUT, mỗi kết quả ứng với một bộ 3 số từ dữ liệu vào được đưa ra trên một dòng.

Ví dụ:

COUNT.INP	COUNT.OUT
3 17 3	
0 20 0	
-1 -1 -1	2 3

Thuật toán:

- **Video hướng dẫn:** <https://youtu.be/CY8zFWzhAo4>
- Đổi số N thành xâu Sn: str(n, Sn); //đổi n thành xâu Sn
- For(i, A, B)
- Begin



Str(i, Si); // đổi i thành xâu Si

If pos(Sn, Si) > 0 then inc(res); // Xâu Sn có xuất hiện trong xâu Si

End;

- Đưa res ra.



g28

```

File Edit Search Run Compile Debug Tools
[ ]
var
  n, i, res, a, b: longint;
  sn, si: string;
begin
  assign(input,'count.inp');reset(input);
  assign(output,'count.out');rewrite(output);
  readln(a, b, n);
  str(n, sn);res := 0;
  for i := a to b do
    begin
      str(i, si);
      if pos(sn, si) > 0 then inc(res);
    end;
  writeln(res);
  close(input);close(output);
end.

```



e47

58) ỦỚC SỐ

Với số nguyên dương N cho trước, ta dễ dàng tìm được tất cả các ước số của nó. Ví dụ, với N = 60, ta có 12 ước số: 1, 2, 3, 4, 5, 6, 12, 15, 20, 30 và 60. Vấn đề ngược lại không dễ dàng: Cho số lượng ước số D hãy tìm số N.

Yêu cầu: Cho số nguyên D ($0 < D \leq 5000$). Tìm số nguyên N nhỏ hơn $10^{15} + 1$ có đúng D ước số.

Dữ liệu: Vào từ file văn bản DIVISOR.INP, gồm nhiều dòng, mỗi dòng chứa một số nguyên D.

Kết quả: Đưa ra file văn bản DIVISOR.OUT các số nguyên tìm được, mỗi số trên một dòng. Trong trường hợp có nhiều số cùng thoả mãn – đưa ra số nhỏ nhất. Nếu không tồn tại số nguyên trong phạm vi đã cho – đưa ra thông báo NO.

Ví dụ:

DIVISOR.INP	DIVISOR.OUT
3	4
4	6
12	60
60	5040
4911	NO



g21

Thuật toán:

- Hàm đếm ước (u)



```
Function demuoc(u: longint) : longint;
Var i, dem: longint;
Begin
  Dem:=0;
  For i:= 1 to trunc(sqrt(u)) do
    If u mod i = 0 then
      Begin
        dem := dem + 1; //nếu tính tổng ước S:=S + i;
        if i <> u div i then
          dem := dem + 1; //nếu tính tổng ước S:= S + u div i;
      End;
    Exit(dem); // exit(s);
End;
```

- **Chương trình chính:**

- $N = 1;$
- Chừng nào $N \leq 1000000$ và $Souoc(N) \neq D$ làm $Inc(N);$
- Nếu $N > 1000000$ thì thông báo “NO”
Ngược lại, đưa N ra.

```
Free Pascal
File Edit Search Run Compile Debug Tools Options Window Help
E:\Free Pascal\bin\ttran\2022\on THCS LHP\dot2\day2\bai15_divisor.pas
const
  fi = 'divisor.inp';
  fo = 'divisor.out';
var d,n : longint;
  function demuoc(u : longint) : longint;
  var r,i : longint;
  begin
    r := 0;
    for i := 1 to trunc(sqrt(u)) do
      if u mod i = 0 then
        begin
          r := r + 1;
          if i <> u div i then
            r := r + 1;
        end;
    exit(r);
  end;
begin
  assign(input,fi); reset(input);
  assign(output,fo); rewrite(output);
  readln(d);
  n := 1;
  while (n<=1000000) and (demuoc(n) <> d) do
    n := n + 1;
  if n > 1000000 then write('NO') else write(n);
  close(input); close(output);
  3:18
F1 Help F2 Save F3 Open Alt+F9 Compile F9 Make Alt+F10 Local menu
```

```

1 #include <bits/stdc++.h>
2 #define nmax 100007
3 using namespace std;
4 int a,b,n;
5 int demuoc(int u)
6 {
7     int dem=0;
8     for(int i=1;i*i<=u;i++)
9         if(u%i==0)
10            {
11                dem++;
12                if(i!=u/i) dem++;
13            }
14     return dem;
15 }
16 int main()
17 {
18     int res=0;
19     int d;
20     cin>>d;
21     int n=1;
22     while(n<=1000000&&demuoc(n)!=d) n++;
23     if(n>1000000) cout<<"NO";
24     else cout<<n;
25     return 0;
26 }
27

```

59) KHÔNG VÀ MỘT

Các số nhị phân và bản đồ phân bố các bit 0, 1 luôn là vấn đề thu hút sự chú ý của các nhà lập trình trong nhiều bài toán khác nhau. Ở đây chúng ta quan tâm đến vấn đề sau: Có bao nhiêu số nhị phân dương thoả mãn các tính chất sau:

- Có đúng N bit và không có các bit 0 ở đầu ($1 < N \leq 64$),
- Số lượng các bit 0 và số lượng các bit 1 bằng nhau,
- Là bội của K ($0 < K \leq 64$).

Dữ liệu: Vào từ file văn bản BIN.INP gồm nhiều dòng, mỗi dòng chứa 2 số nguyên dương N K.

Kết quả: Đưa ra file văn bản BIN.OUT các số lượng tìm được, mỗi giá trị trên một dòng.

Ví dụ:

Giải thích: Với các N và K cho trước, bản thân các số nhị phân thoả mãn sẽ là:

N = 6, K = 3	N = 6, K = 4	N = 6, K = 2
101010	111000 110100	111000 110100



	101100	101100 110010 101010 100110
--	---------------	------------------------------------------------------------------



g22

60) ĐA HOÁN VỊ NGUYÊN TỐ

Hoán vị dãy các chữ số hệ 10 có một tính chất thú vị: hiệu của 2 hoán vị luôn là một số chia hết cho 9. Thật là bất ngờ phải không? Ví dụ: $|458967 - 456879| = 2088 = 9 * 232$

856213	638125	365128
862315	156832	165328
531682	618325	632518
361852	685312	612538
356821	615823	562318
163852	682531	623851
653182	136285	561283

Không yêu cầu bạn phải chứng minh tính chất này (mặc dù là rất dễ) mà đề nghị bạn lưu ý đến một khía cạnh khác của giá trị tuyệt đối hiệu này. Có những số mà hiệu giữa nó và một hoán vị của nó có dạng $9p$, trong đó p là số nguyên tố nhỏ hơn 1 111 111. Những số này gọi là hoán vị nguyên tố. Ví dụ, $92 - 29 = 63 = 9*7$. 7 là số nguyên tố, vì vậy 92 được gọi là hoán vị nguyên tố.

Yêu cầu: Cho 2 số nguyên dương A và B ($0 < A, B \leq 99\ 999\ 999$, $|A-B| \leq 1\ 000$). Hãy xác định khoảng $[A, B]$ chứa bao nhiêu hoán vị nguyên tố.

Dữ liệu: Vào từ file văn bản PRIME_P.INP:

- Dòng đầu tiên chứa số nguyên T - số lượng Tests,
- T dòng sau: mỗi dòng chứa 2 số nguyên A B.

Kết quả: Đưa ra file văn bản PRIME_P.OUT: ứng với mỗi cặp A, B đưa ra số lượng hoán vị nguyên tố tìm được, mỗi kết quả trên một dòng.

Ví dụ:

PRIME_P.INP	PRIME_P.OUT
2	0
1 10	5
1 20	



g23



Thuật toán:

- **Video hướng dẫn:** <https://youtu.be/SPxTiq2vZtk>
- Tô chức hàm nguyên tố (u).
- Tô chức hàm Phantich(i) – tách các chữ số của i lưu vào mảng a có K phần tử.

```
Procedure phantich(u: longint);
Var i, tam: longint;
Begin
  K:= 0;
  While u > 0 do
    Begin
      Inc(k);
      A[k]:= U mod 10;
      U:= u div 10;
    End;
    // đảo ngược mảng a có K phần tử.
  For i:=1 to k div 2 do
    Begin
      Tam:= a[i];
      A[i]:= a[k-i+1];
      A[k-i+1] := tam;
    End;
  End;
```



- Tô chức hàm **ghepso()** – ghép mảng hoán vị thành một số nguyên

```
Function ghepso(): longint;
Var i, s: longint;
Begin
  S:= 0;
  For i:= 1 to k do
    S:= s* 10 + a[b[i]];
  Exit(s);
End;
```

- **Thủ tục xuli()**

```
Procedure xuli()
Var tam: longint;
Begin
  Tam:= ghepso();
  If Nguyento(Abs(N-tam) div 9) = true then res:= res + 1;
End;
```

- **Giải thuật sinh hoán vị**

```
Procedure thu(i: longint);
Var j : longint;
Begin
```



```
For j:=1 to k do
  If fre[j] = true then
    Begin
      B[i] := j;
      If i=k then xuli()
      Else
        Begin
          Fre[j]:= false;
          Thu(i+1);
          Fre[j] := true;
        End;
    End;
  End;
```

- Chương trình chính:

```
Begin
  Res :=0;
  For (i, A, B)
    Begin
      Phantich(i); // phân tích i thành mảng a có k phần tử
      N:= i;
      For(p,1, k) fre[p] := true;
      Thu(1);
    End;
  Write(res); // đưa số lượng hoán vị nguyên tố ra.
End.
```

```
[ ] var
  a, b, i, p, k, res, n: longint;
  c, d: array[-7..100007] of longint;
  fre: array[-7..100007] of boolean;
function snt(u: longint): boolean;
var j: longint;
begin
  if u <= 1 then exit(false);
  for j := 2 to trunc(sqrt(u)) do
    if u mod j = 0 then exit(false);
  exit(true);
end;
```



```
procedure pt(u: longint);
var i, tam: longint;
begin
  k := 0;
  while u > 0 do
    begin
      inc(k);
      d[k] := u mod 10;
      u := u div 10;
    end;
  for i := 1 to k div 2 do
    begin
      tam := d[i];
      d[i] := d[k-i+1];
      d[k-i+1] := tam;
    end;
end;
function gs(): longint;
var i, s: longint;
begin
  s := 0;
  for i := 1 to k do s := s * 10 + d[c[i]];
  exit(s);
end;
procedure xuli();
var tam: longint;
begin
  tam := gs();
  if snt(abs(n - tam) div 9) = true then inc(res);
end;
procedure thu(i: longint);
var j: longint;
begin
  for j := 1 to k do
    if fre[j] = true then
      begin
        c[i] := j;
        if i = k then xuli
        else
          begin
            fre[j] := false;
            thu(i + 1);
            fre[j]:= true;
          end;
      end;
end;
end;
```



```
begin
  assign(input,'prime.inp');reset(input);
  assign(output,'prime.out');rewrite(output);
  readln(a, b);
  res := 0;k := abs(a-b);
  for i := a to b do
    begin
      n := i;
      pt(i);
      for p := 1 to k do fre[p] := true;
      thu(1);
    end;
  writeln(res);
  close(input);close(output);
end.
```



```
1 #include<bits/stdc++.h>
2 #define nmax 100007
3 using namespace std;
4 int n,res,k;
5 int a[nmax],b[nmax];
6 int c[nmax],fre[nmax],m[nmax];
7 bool snt(int u)
8 {
9     if(u<=1) return false;
10    if(u==2||u==3) return true;
11    for(int j=2;j*j<=u;j++)
12        if(u%j==0) return false;
13    return true;
14 }
15 void phantich(int u)
16 {
17     k=0;
18     while(u>0)
19     {
20         k++;
21         a[k]=u%10;
22         u=u/10;
23     }
24     for(int i=1;i<=k/2;i++)
25         swap(a[i],a[k-i+1]);
26 }
27 int ghepso()
28 {
29     int s;
30     s=0;
31     for(int i=1;i<=k;i++)
32         s=s*10+a[b[i]];
33     return s;
34 }
35 void x1()
36 {
37     int tam;
38     tam=ghepso();
39     if(snt(abs(n-tam)/9)) res++;
40 }
41 void thu(int i)
42 {
43     for(int j=1;j<=k;j++)
44         if(fre[j])
45         {
46             b[i]=j;
47             if(i==k) x1();
48             else
49             {
50                 fre[j]=false;
51                 thu(i+1);
52                 fre[j]=true;
53             }
54         }
55 }
56 int main()
57 {
58     freopen("PRIME_P.inp","r",stdin);
59     freopen("PRIME_P.out","w",stdout);
60     int A,B;
61     cin>>A>>B;
62     res=0;
63     for(int i=A;i<=B;i++)
64     {
65         n=i;
66         phantich(i);
67         //for(int p=1;p<=k;p++) cout<<a[p]<<" ";
68         //cout<<endl;
69         for(int p=1;p<=k;p++) fre[p]=true;
70         thu(1);
71     }
72     cout<<res;
73     return 0;
74 }
75 //Code by Quân ^^
```



61) QUÂN XE

123



Xét bàn cờ kích thước vô hạn, các cột được đánh số từ trái sang phải từ $-\infty$ đến $+\infty$, tương tự như vậy, các hàng được đánh số từ dưới lên trên. Trên bàn cờ có một số ô bị cắt bỏ. Tại một ô (không bị cắt) có một quân xe. Bằng một bước đi quân xe có thể di chuyển tới ô bất kỳ cùng hàng hoặc cùng cột nếu giữa 2 ô này không có ô bị cắt.

Yêu cầu: xác định số nước đi tối thiểu từ ô ban đầu đến ô đích nếu điều đó là có thể được.

Dữ liệu: Vào từ file văn bản ROOK.INP:

- Dòng đầu tiên chứa 4 số nguyên $X_1 \ Y_1 \ X_2 \ Y_2$, trong đó (X_1, Y_1) - toạ độ ô xuất phát, (X_2, Y_2) toạ độ ô đích, các toạ độ nằm trong phạm vi từ -10^9 đến 10^9 .
- Dòng thứ 2 chứa số nguyên N - số ô bị cắt ($0 < N \leq 1000$),
- N dòng sau: mỗi dòng chứa một cặp số nguyên $P \ Q$ xác định toạ độ ô bị cắt, $-10^9 \leq P, Q \leq 10^9$.

Kết quả: Đưa ra file văn bản ROOK.OUT số nước đi tối thiểu hoặc -1 nếu không thể đi được.

Ví dụ:

ROOK.INP
1 -1 5 -4
4
1 1
5 -5
2 -4
4 -1

ROOK.OUT
3



e5

62) CÂN ĐĨA

Người ta dùng cân đĩa để cân vật có khối lượng nguyên M ($1 \leq M \leq 10^{100}$). Các quả cân có khối lượng 3^k , $k = 0, 1, 2, \dots$. Ứng với mỗi k chỉ có một quả cân khối lượng 3^k . Khối lượng cần cân để ở đĩa bên trái, các quả cân có thể để ở cả hai đĩa phải và trái.

Yêu cầu: Với M cho trước, hãy chỉ ra cách đặt các quả cân để có thể cân được khối lượng M .

Dữ liệu: Vào từ file văn bản SCALES.INP, chứa n một số nguyên M .

Kết quả: Đưa ra file văn bản SCALES.OUT 2 dòng:

- Dòng thứ nhất chỉ ra các quả cân để ở đĩa bên trái hoặc số 0 nếu không cần đặt,
- Dòng thứ 2: chỉ ra các quả cân ở đĩa bên phải.

Ví dụ:

SCALES.INP
42

SCALES.OUT
3 3 9 27
1 81

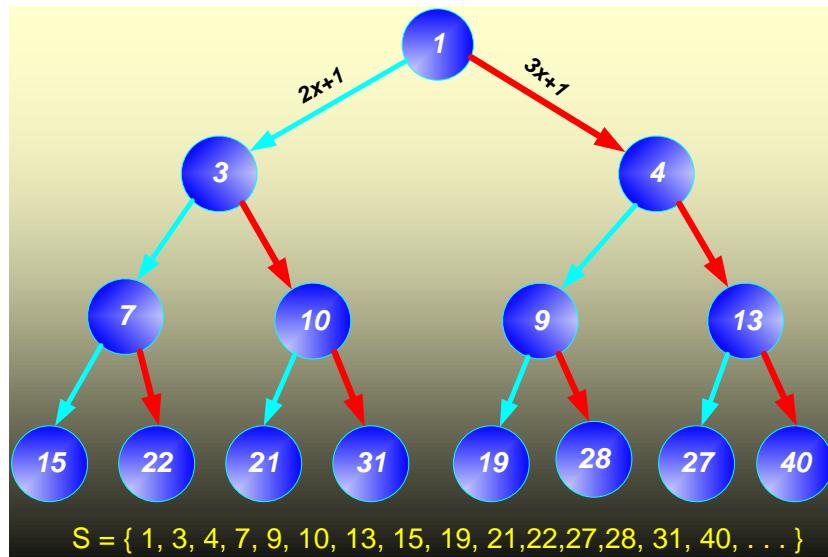


63) TẬP SỐ

Xét tập số nguyên S được định nghĩa như sau:



- $1 \in S$,
- Nếu $x \in S$, thì $2x+1$ và $3x+1$ cũng thuộc S ,
- S không chứa số nào khác ngoài các số theo quy tắc đã nêu.



Yêu cầu: Cho số nguyên N ($1 \leq N \leq 2^{31}-1$). Hãy xác định số thứ N của S , nếu các số trong S được sắp xếp theo thứ tự tăng dần.

Dữ liệu: Vào từ file văn bản SET.INP gồm nhiều dòng, mỗi dòng chứa một số nguyên N .

Kết quả: Đưa ra file văn bản SET.OUT, mỗi kết quả là một số nguyên và đưa ra trên một dòng.

Ví dụ:

SET.INP	SET.OUT
100	418
254	1461



Thuật toán:

- Thủ tục sắp xếp dãy số: sapxep().
- Hàm timson(u: longint): longint; trả về phần tử thứ N của dãy a.
 - Xây dựng mảng a có N phần tử: $a[1] = 1$; $i := 1$;
 - Chừng nào $i < N$ làm

Inc(i); $a[i] := 2*a[i-1] + 1$;

If $i = N$ then break;

Inc(i); $a[i] := 3*a[i-1] + 1$;



- Sapxep(1,N);
- Exit(a[n]);
- Chương trình chính: write(timson(N));

64) SỐ RỖ RÀNG

Các nhà toán học đưa vào lý thuyết nhiều cách phân loại số, ví dụ, với các số nguyên ta có số chẵn và số lẻ, số nguyên tố và hợp số, số chính phương và không chính phương v. v. . . Bob cũng muốn đặt dấu ấn của mình trong lĩnh vực phân loại số. Bob chia các số nguyên dương thành 2 loại: *rõ ràng* và *luẩn quẩn*. Việc xác định một số thuộc loại nào được thực hiện theo giải thuật sau: Với số nguyên dương n , ta tạo số mới bằng cách lấy *tổng bình phương các chữ số* của nó, với số mới này ta lại lặp lại công việc trên. Nếu trong quá trình trên, ta nhận được số mới là 1, thì số n ban đầu được gọi là số rõ ràng. Ví dụ, với $n = 19$, ta có:

$$19 \rightarrow 82 (= 1^2 + 8^2) \rightarrow 68 \rightarrow 100 \rightarrow 1$$

Như vậy, 19 là số rõ ràng.

Không phải mọi số đều rõ ràng. Ví dụ, với $n = 12$, ta có:

$$12 \rightarrow 5 \rightarrow 25 \rightarrow 29 \rightarrow 85 \rightarrow 89 \rightarrow 145 \rightarrow 42 \rightarrow 20 \rightarrow 4 \rightarrow 16 \rightarrow 37 \rightarrow 58 \rightarrow 89 \rightarrow 145$$

Rất thú vị với cách phân loại của mình, Bob muốn biết, trong thực tế, số rõ ràng có nhiều hay ít.

Yêu cầu: Cho hai số nguyên dương A và B ($1 \leq A \leq B \leq 10\,000\,000$). Hãy xác định K – số lượng số rõ ràng nằm trong khoảng [A, B].

Dữ liệu: Vào từ file văn bản CLEAR.INP gồm một dòng chứa 2 số nguyên A B.

Kết quả: Đưa ra file văn bản CLEAR.OUT số nguyên K.

Ví dụ:

CLEAR.INP	CLEAR.OUT
2 20	4



Thuật toán:

- **Video hướng dẫn:** https://youtu.be/7V4we_g7XoA
- Hàm tính tổng bình phương các chữ số của u: function **tongbp**(u: longint): longint;

```

Function tongbp(u: longint): longint;
Var s: longint = 0;
Begin
  While u > 0 do
    Begin
      S:= s+ sqr(U mod 10);
      U:= u div 10;
    End;
  
```



```
Exit(s);  
End;
```

- Hàm kiểm tra u có là số rõ ràng:

```
Function sorr(u: longint): boolean;  
Var c: array[1..100000] of boolean;  
    I,: longint;  
Begin  
    For(i,1,100000) c[i]:= false;  
    While true do  
        Begin  
            U:= tongbp(u);  
            If c[u] = true then exit(false);  
            If u = 1 then exit(true);  
            C[u]:= true;  
        End;  
    End;
```

$N = 22, 8, 64, 52, 29, 85, 89, 145, 42, 20, 4, 16, 37, 58, 89, ;$

- Chương trình chính:

- For(i, a, b)
 - Nếu sorr(i) = true thì inc(res);
 - Đưa res ra.



```
1 #include <bits/stdc++.h>
2 #define nmax 100007
3 using namespace std;
4 int tongbp(int k)
5 {
6     int s=0;
7     while(k>0)
8     {
9         s=s+(k%10)*(k%10);
10        k=k/10;
11    }
12    return s;
13 }
14 bool sorr(int u)
15 {
16     bool c[nmax];
17     for(int i=1;i<=100007;i++) c[i]=false;
18     while(true)
19     {
20         u=tongbp(u);
21         if(c[u]) return false;
22         if(u==1) return true;
23         c[u]=true;
24     }
25 }
26 int main()
27 {
28     int a,b;
29     int res=0;
30     cin>>a>>b;
31     for(int i=a;i<=b;i++)
32         if(sorr(i)) res++;
33     cout<<res;
34     return 0;
35 }
36 }
```

Free Pascal IDE

```

File Edit Search Run Compile Debug Tools
[ ] var
  n, ii, res, a, b: longint;
function tbp(u: longint): longint;
var s: longint = 0;
begin
  while u > 0 do
    begin
      s := s + sqr(u mod 10);
      u := u div 10;
    end;
  exit(s);
end;
function srr(u: longint): boolean;
var c: array[-7..1000007] of boolean;
  i: longint;
begin
  for i := 1 to 100000 do c[i] := false;
  while true do
    begin
      u := tbp(u);
      if c[u] then exit(false);
      if u = 1 then exit(true);
      c[u] := true;
    end;
end;
begin
  assign(input,'clear.inp');reset(input);
  assign(output,'clear.out');rewrite(output);
  readln(a, b);
  res := 0;
  for ii := a to b do
    if srr(ii) then inc(res);
  writeln(res);
  close(input);close(output);
end.

```

65) DÃY SỐ

Cho một nhóm các số nguyên tố. Ta có thể xây dựng một dãy số nguyên dương tăng dần lớn hơn 1 mà trong kết quả phân tích ra thừa số nguyên tố mỗi số trong dãy chỉ chứa các số nguyên tố đã cho. Ví dụ, cho nhóm số nguyên tố 2, 3, 5, thì dãy số cần xây dựng sẽ là 2, 3, 4, 5, 6, 8, 9, 10, 12, 15, ...

Yêu cầu: Cho n , các số nguyên tố p_i , $i = 1 \div n$ ($1 \leq n \leq 1\,000$, $p_i \leq 10\,000$) và số nguyên j ($1 \leq j \leq 2^{31}$). Các số nguyên tố được cho theo thứ tự tăng dần. Hãy tìm và đưa ra số thứ j trong dãy nêu trên.



Dữ liệu: Vào từ file văn bản SEQUENCE.INP:

- Dòng đầu tiên chứa số nguyên n ,
- Dòng thứ $i+1$ chứa số p_i ,
- Dòng cuối cùng chứa số nguyên j .

Kết quả: Đưa ra file văn bản SEQUENCE.OUT số nguyên tìm được.

Ví dụ:

SEQUENCE.INP	SEQUENCE.OUT
3	
2	
3	
5	
1000	

i4

Thuật toán:

- **Video hướng dẫn:** https://youtu.be/rT_cZgwJmuI
- Hàm kiểm tra số u khi phân tích thành tích các thừa số nguyên tố chỉ bao gồm các SNT đã cho hay không? Function **kiemtra**(u: longint):boolean;

```
Function kiemtra(u: longint):boolean;
Var i: longint;
Begin
  If i = 1 then exit(false);
  i:=2;
  while i <= trunc(sqrt(u)) do
    If u mod i = 0 then
      Begin
        If c[i] = false then exit(false);
        U:= u div i;
      End
    Else inc(i);
  If u > 1 then
    If c[u] = false then exit (false);
  Exit(true);
End;
```

- Hàm tìm phần tử thứ u của dãy số: function **timson**(u: longint): longint;

```
function timson(u: longint): longint;
var i, k: longint;
d: array[1..100000] of longint;
begin
  i:= 1; k:= 0; //k lưu số lượng các phần tử thỏa mãn
  while k < u do
    begin
      while kiemtra(i) = false do inc(i);
      inc(k);
      d[k]:= i;
```



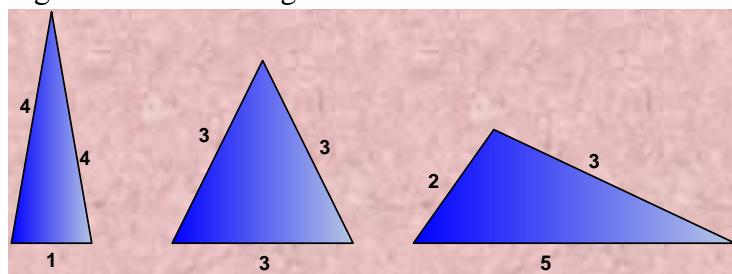
```
inc(i);  
end;  
exit(d[u]);  
end;
```

- **Chương trình chính:**

- Khởi tạo mảng C bởi false: for(i,1,100000) c[i]:= false;
- Readln(n);
- For(i,1,N)
Begin
 Readln(x);
 C[x] := true; //đánh dấu số nguyên tố x có xuất hiện
End;
- Readln(j);
- Write(timson(j));

66) TAM GIÁC

Xét các tam giác có diện tích lớn hơn 0 và độ dài các cạnh đều là nguyên. Với một số nguyên P cho trước ($P \geq 3$) có thể có 1 hoặc nhiều tam giác không bằng nhau từng đôi một và cùng có chu vi là P. Ví dụ, với $P = 9$ ta có 3 tam giác khác nhau cùng có chu vi là 9.



Yêu cầu: Cho P ($3 \leq P \leq 2^{31}-1$). Hãy xác định N - số tam giác không bằng nhau có cạnh nguyên và chu vi là P.

Dữ liệu: Vào từ file văn bản TRIANGLE.INP gồm nhiều dòng, mỗi dòng chứa một số nguyên P.

Kết quả: Đưa ra file văn bản TRIANGLE.OUT các số N tìm được. Mỗi số trên một dòng.

Ví dụ:

TRIANGLE.INP
5
7
9
320920391

TRIANGLE.OUT
1
2
3
2145622901773234



67) CẶP SỐ 0

Xuất phát từ xâu S ban đầu chỉ chứa một ký tự ‘1’, người ta biến đổi n lần theo quy tắc sau:

- Tạo xâu T bằng cách đảo các ký tự trong S : ‘1’ thành ‘0’ và ngược lại,
- Tính S mới: $S := T + S$.

Với cách biến đổi đó, ta có:

n	S
1	01
2	1001
3	01101001

Yêu cầu: Cho biết n ($0 < n \leq 1\ 000$). Hãy xác định cặp số 0 trong xâu S sau n lần biến đổi.

Dữ liệu: Vào từ file văn bản PAIR.INP gồm nhiều dòng, mỗi dòng chứa một số nguyên n .

Kết quả: Đưa ra file văn bản PAIR.OUT, mỗi kết quả đưa ra trên một dòng dưới dạng số nguyên.

Ví dụ:

PAIR.INP
2
3

PAIR.OUT
1
1



68) GHÉP SỐ

Cho n số nguyên dương a_1, a_2, \dots, a_n ($1 < n \leq 50$), mỗi số không vượt quá $2\ 147\ 483\ 647$. Từ các số này người ta tạo ra một số nguyên mới bằng cách ghép tất cả các số đã cho, tức là viết liên tiếp các số đã cho với nhau. Ví dụ, với $n = 4$ và các số 123, 124, 56, 90 ta có thể tạo ra các số mới – 1231245690, 1241235690, 5612312490, 9012312456, 9056124123, v. v... Có thể dễ dàng thấy rằng, với $n = 4$, ta có thể tạo ra 24 số mới. Trong trường hợp này, số lớn nhất có thể tạo ra là 9056124123.

Yêu cầu: Cho n và các số a_1, a_2, \dots, a_n . Hãy xác định số lớn nhất có thể tạo ra khi ghép các số đã cho thành một số mới.

Dữ liệu: Vào từ file văn bản NUMJOIN.INP, gồm nhiều tests, mỗi test ghi trên 2 dòng:

- Dòng thứ nhất chứa số nguyên n ,
- Dòng thứ 2 chứa n số nguyên $a_1 \ a_2 \ \dots \ a_n$.

Dữ liệu kết thúc bằng dòng chứa một số 0.



Kết quả: Đưa ra file văn bản NUMJOIN.OUT, mỗi kết quả đưa ra trên một dòng dưới dạng một số nguyên.

Ví dụ:

NUMJOIN.INP
4
123 124 56 90
5
123 124 56 90 9
5
9 9 9 9 9
0

NUMJOIN.OUT
9056124123
99056124123
99999



i27

69) SỐ NGUỒN

Giả thiết N là số nguyên dương. Số nguyên M là tổng của N với các chữ số của nó. N được gọi là nguồn của M. Ví dụ, N = 245, khi đó M = 245 + 2 + 4 + 5 = 256. Như vậy, nguồn của 256 là 245.

Không có gì đáng ngạc nhiên nếu thấy rằng có những số không có nguồn và có số lại có nhiều nguồn. Ví dụ, số 216 có 2 nguồn là 198 và 207.

Yêu cầu: Cho số nguyên M. hãy tìm nguồn nhỏ nhất của nó. Nếu M không có nguồn thì đưa ra số 0.

Dữ liệu: Vào từ file văn bản GEN.INP :

- Dòng đầu tiên chứa số nguyên T – số lượng Tests,
- T dòng sau: mỗi dòng chứa một số nguyên M.

Kết quả: Đưa ra file văn bản GEN.OUT, mỗi kết quả đưa ra trên một dòng.

Ví dụ:

GEN.INP
3
216
121
2005

GEN.OUT
198
0
1979



i37

70) SỐ PINARY

Pinary là số nguyên chứa các số 0 và 1 thoả mãn các điều kiện:

- ❖ Không bắt đầu bằng 0,
- ❖ Không chứa 2 số 1 liên tiếp nhau.



Ví dụ, các số sau là số Pinary: 1, 10, 100, 101, 1000, 1001, 1010, ... Còn các số sau không phải là Pinary: 00101, 1001101.

Các số Pinary được sắp xếp theo thứ tự từ điển.

Yêu cầu: Cho số nguyên n . Hãy xác định số Pinary thứ n .

Dữ liệu: Vào từ file văn bản PINARY.INP gồm nhiều dòng, mỗi dòng chứa một số nguyên n ($0 < n \leq 90\,000\,000$).

Kết quả: Đưa ra file văn bản PINARY.OUT các số Pinary tìm được, mỗi số trên một dòng.

Ví dụ:

PINARY.INP
7
2000
22

PINARY.OUT
1010
1001000001001000
100001



71) CHỌN HÌNH

Cho lưới ô vuông kích thước m dòng và n cột, các dòng được đánh số từ 1 tới m từ trên xuống dưới, các cột được đánh số từ 1 đến n từ trái sang phải. Ở mỗi ô (i, j) có ghi một số nguyên a_{ij} ($2 \leq m, n \leq 500$, $|a_{ij}| \leq 10^7$). Phải lựa chọn 4 ô, sao cho tâm của 4 ô này sẽ là đỉnh của một hình vuông có cạnh song song với cạnh của lưới và tổng các số trong 4 ô đó là lớn nhất.

Dữ liệu: Vào từ file văn bản CHOOSE.INP:

- Dòng đầu tiên chứa 2 số nguyên m n ,
- m dòng sau: mỗi dòng chứa n số nguyên mô tả một dòng của lưới.

$m = 5$	1	1	1	1	1
	1	2	-1	1	1
	1	1	1	1	1
	1	1	-1	3	1
	1	1	1	1	1

$n = 5$

Kết quả: Đưa ra file văn bản CHOOSE.OUT:

- Dòng thứ nhất chứa số nguyên r – tổng lớn nhất tìm được,
- Dòng thứ 2 chứa 4 số nguyên x_1 y_1 x_2 y_2 – toạ độ góc trên trái và dưới phải của các ô được chọn ($1 \leq x_1 < x_2 \leq m$, $1 \leq y_1 < y_2 \leq n$).

Nếu có nhiều lời giải thì chỉ cần đưa ra một nghiệm bất kỳ.

Ví dụ:

CHOOSE.INP
5 5
1 1 1 1 1
1 2 1 1 1
1 1 1 1 1

CHOOSE.OUT
7
2 2 4 4



1 1 1 3 1
1 1 1 1 1
5 5
1 -1 -1 -1 -1
-1 -2 -1 -1 -1
-1 -1 -1 -1 -1
-1 -1 -1 -3 -1
-1 -1 -1 1 -1

0
1 1 5 4



i45

---Hết---