# Các thuật toán trên tập số

## Một số thư viện tiện ích

```
#include <iostream> // keyboard input / output
#include <fstream> // disk (file) input / output
#include <bitset> // bit array
#include <cmath> // mathematic functions
#include <math.h> // mathematic functions
#include <windows.h> // windows environment
#include <cstring>
#include <string>
#include <bits/stdc++.h>
#include <time.h>
#include <algorithm>
```

### *Lật số nguyên trong hệ thập phân*

```
Algorithm Rev(x) → y

Lấy từng chữ số hàng đơn (đầu trái) của x
ghép vào đầu phải y.
Chữ số hàng đơn của x = x % 10.
Phần còn lại của x: x  = x / 10.
```

```
typedef long long Long; // 64 bit

Long Rev(Long x) {
   int y = 0;
   while (x != 0) {
    y = y * 10 + (x % 10);
    x /= 10;
   }
   return y;
}
```

### *Độ phức tạp tính toán tuyến tính theo Len(x)*

### *Lật số nguyên trong hệ đếm base*

```
Algorithm Rev(x, base = 10) → y

Lấy từng chữ số hàng đơn (đầu trái) của x
ghép vào đầu phải y.
Chữ số hàng đơn của x = x % base.
Phần còn lại của x: x  = x / base.
```

```
Long Rev(Long x, int base = 10) {
   Long y = 0;
   while (x != 0) {
    y = y * base + (x % base);
    x /= base;
   }
   return y;
}
```

*Độ phức tạp tính toán tuyến tính theo Len(x)*

*Số đối xứng (palindrome)*

Definition

```
x là số palindrome trong hệ đếm a khi và chỉ khi
x = Rev(x, a).
```

```
Algorithm Pal(x, base)

return Rev(x,base) == x
```

```
bool Pal(Long x, int base = 10) {
   return x == Rev(x, base);
}
```

*Độ phức tạp tính toán tuyến tính theo Len(x)*

*Số đối xứng hai hệ đếm a, b (Bipalindrome)*

Definition

```
x là số palindrome trong hai hệ đếm a và b
khi và chỉ khi Pal(x,a) && Pal(x,b).
```

```
Algorithm BiPal(x, a, b)

return Pal(x, a) && Pal(x, b)
```

```
bool BiPal(Long x, int a, int b) {
   if (x == Rev(x, a)) {
     return (x == Rev(x,b));
  }
  return false;
}
```

*Độ phức tạp tính toán tuyến tính theo Len(x)*

*Biểu diễn số dưới dạng string*

```
Algorithm IntToStr(x) → s

Lấy từng chữ số hàng đơn (đầu trái) của x
ghép vào đầu trái string s.
Chữ số hàng đơn của x = x % base.
Phần còn lại của x: x  = x / base.
```

```
const string DIGIT = "0123456789";
const string alpha = "abcdefghijklmnopqrstuvwxyz";
const string EDIGIT = DIGIT + alpha; // Expanded Digit

typedef long long Long;

string IntToStr(Long x, int base = 10) {
  string s = "";
  if (x == 0) return "0";
  while(x) {
    s = EDIGIT[x % base] + s;
    x /= base;
  }
  return s;
}
```

*Độ phức tạp tính toán tuyến tính theo Len(x)*

## Problem 1. Bi Palindrome

Liệt kê các số đối xứng trong khoảng [1 : 1M] với hai hệ đếm 10 và 16 đồng thời.

*Độ phức tạp tính toán tuyến tính theo n =1M*

Program

```
#include <iostream>

using namespace std;

void Go() {
   cout << " ? ";
   fflush(stdin);
   if (cin.get() == '.') exit(0);
}

#include <iostream>

using namespace std;

typedef long long Long;

const string DIGIT = "0123456789";
```

```
const string alpha = "abcdefghijklmnopqrstuvwxyz";
const string EDIGIT = DIGIT + alpha; // Expanded Digit

Long Rev(Long x, int base = 10) {
   Long y = 0;
   while (x != 0) {
    y = y * base + (x % base);
    x /= base;
   }
   return y;
}

bool Pal(Long x, int base = 10) {
   return x == Rev(x, base);
}

bool BiPal(Long x, int base1, int base2) {
   if (x == Rev(x, base1)) {
     return (x == Rev(x, base2));
   }
   return false;
}

string IntToStr(Long x, int base = 10) {
  string s = "";
  if (x == 0) return "0";
  while(x) {
    s = EDIGIT[x % base] + s;
    x /= base;
  }
  return s;
}

void Run(Long start, Long end, int base1, int base2) {
  int c = 0;
  for (Long x = start; x <= end; ++x) {
     if (BiPal(x, base1, base2)) {
        ++c;
        cout << "\n " << c << ". " << x << " : "
             << IntToStr(x, base1) << "[" << base1 << "],"
             << " " << IntToStr(x,base2) << "[" << base2 << "]";
      }
  }
  cout << "\n Total " << c;
}

main() {
  Run(1, 1000000, 10, 16);
  cout << "\n T h e   E n d";
  return 0;
}
```

## Result

```
 1. 1 : 1[10], 1[16]
 2. 2 : 2[10], 2[16]
```

```
 3. 3 : 3[10], 3[16]
 4. 4 : 4[10], 4[16]
 5. 5 : 5[10], 5[16]
 6. 6 : 6[10], 6[16]
 7. 7 : 7[10], 7[16]
 8. 8 : 8[10], 8[16]
 9. 9 : 9[10], 9[16]
 10. 11 : 11[10], b[16]
 11. 353 : 353[10], 161[16]
 12. 626 : 626[10], 272[16]
 13. 787 : 787[10], 313[16]
 14. 979 : 979[10], 3d3[16]
 15. 1991 : 1991[10], 7c7[16]
 16. 3003 : 3003[10], bbb[16]
 17. 39593 : 39593[10], 9aa9[16]
 18. 41514 : 41514[10], a22a[16]
 19. 90209 : 90209[10], 16061[16]
 20. 94049 : 94049[10], 16f61[16]
 21. 96369 : 96369[10], 17871[16]
 22. 98689 : 98689[10], 18181[16]
 23. 333333 : 333333[10], 51615[16]
 24. 512215 : 512215[10], 7d0d7[16]
 25. 666666 : 666666[10], a2c2a[16]
 26. 749947 : 749947[10], b717b[16]
 27. 845548 : 845548[10], ce6ec[16]
 Total 27
 T h e   E n d
```

## *Chuyển đổi số có dấu*

### Program

```
#include <iostream>

using namespace std;

typedef long long Long;

const string DIGIT = "0123456789";
const string alpha = "abcdefghijklmnopqrstuvwxyz";
const string EDIGIT = DIGIT + alpha; // Expanded Digit

void Go() {
   cout << " ? ";
   fflush(stdin);
   if (cin.get() == '.') exit(0);
}

// -1234 -> "-1234"
string IntToStr(Long x, int base = 10) {
  string s = "";
  if (x == 0) return "0";
  string sign = "";
  if (x < 0) {
     sign = "-";
     x = -x;
  }
```

```
  while(x) {
    s = EDIGIT[x % base] + s;
    x /= base;
  }
  return sign + s;
}

// "-1234" -> -1234
Long StrToInt(string & s, int base = 10) {
  Long val = 0;
  int sign = 1;
 int start = 0;
 if (s[0] == '-') {
    sign = -1;
    start = 1;
  }
 for(int i = start; i < s.length(); ++i) {
    val = val*base + (s[i] - '0');
 }
  return sign*val;
}

void Run() {
  string s;
  int base = 10;
  Long y;
  int er = 0;
  for (int x = -1000; x <= 1020; ++x) {
     s = IntToStr(x, base);
     y = StrToInt(s,base);
     cout << "\n " << x << " -> " << s << " -> " << y;
     if (x == y) cout << " CORRECT.";
     else {
            cout << " ERROR.";
            ++er;
          }
  }
  cout << "\n Total " << er << " error(s.)";
}

main() {
  Run();
  cout << "\n T h e   E n d";
  return 0;
}
```

## Problem 2. 5-Palindrome

Trong hệ thập phân có bao nhiêu số đối xứng 5 chữ số?

### Phương án 1. Programming

```
#include <iostream>

using namespace std;

int Rev(int x) {
    int y = 0;
```

```
    while(x) {
       y = y * 10 + (x % 10);
       x /= 10;
    }
    return y;
}

bool Pal(int x) {
  return Rev(x) == x;
}

int Run() {
   int c = 0;
   for (int x = 10000; x <= 99999; ++x) {
      if (Pal(x)) {
         ++c;
         cout << "\n " << x;
      }
   }
   cout << "\n Total " << c; // 900
}

main() {
  Run();
  cout << "\n T h e   E n d";
  return 0;
}
```

## Phương án 2. Gen Programming

Thay vì tìm kiếm, hãy tạo ra đối tượng!

```
#include <iostream>

using namespace std;

int Run() {
   int c = 0;
   int z;
   for (int x = 10; x <= 99; ++x) {
      for (int y = 0; y <= 9; ++y) {
        // z = xyx
       z = x*1000 + (y*100 + x);
         ++c;
         cout << "\n " << z;
      }
   }
   cout << "\n Total " << c;
}

main() {
  Run();
  cout << "\n T h e   E n d";
  return 0;
}
```

### Phương án 3. Tính nhẩm

Số đối xứng 5 chữ số có dạng abcba, a > 0

Vậy ab = 10 .. 99, c = 0..9.

Với mỗi số ab từ 10 đến 99 ta thêm lần lượt các số c = 0 đến 9 rồi ghép với (ab)' là thu được 10 số đối xừng.

Thí dụ, ab = 18, (ab)' = 81

18081, 18181,18281, 18381, 18481,18581 , 18681, 18781,18881, 18981.

Có 90 số dạng ab do đó tổng cộng có 90*10 = 900 số palindrome 5 chữ số.

## Problem 3. Độ cao của số

Độ cao (Height) của số nguyên không âm là tổng các chữ số của số đó.

H(123) = 1 + 2 + 3 = 6, H(2021) = 2 + 0 + 2 + 1 = 5, H(0) = 0.

Liệt kê các số tự nhiên trong khoảng $0 \ldots 2^9$ chứa đúng 5 bit 1 dưới dạng nhị phân.

### Phương án 1. Programming

```
#include <iostream>

using namespace std;

typedef long long Long;

const string DIGIT = "0123456789";
const string alpha = "abcdefghijklmnopqrstuvwxyz";
const string EDIGIT = DIGIT + alpha; // Expanded Digit

void Go() {
   cout << " ? ";
   fflush(stdin);
   if (cin.get() == '.') exit(0);
}

int H(Long x, int base = 10) {
  int sum = 0;
  while (x) {
     sum += x % base;
     x /= base;
  }
  return sum;
}

// -1234 -> "-1234"
string IntToStr(Long x, int base = 10) {
  string s = "";
  if (x == 0) return "0";
  string sign = "";
  if (x < 0) {
     sign = "-";
     x = -x;
```

```
 }
  while(x) {
    s = EDIGIT[x % base] + s;
    x /= base;
  }
  return sign + s;
}


int Run(int m, int k) {
   int v = 1 << m; // v = 2^9
   int c = 0;
   for (int x = 0; x <= v; ++x) {
      if (H(x, 2) == k) {
         ++c;
         cout << "\n " << c << ". " << x << " " << IntToStr(x,2); // Go();
       }
   }
   cout << "\n Total " << c;
}

main() {
  Run(9, 5);
  cout << "\n T h e   E n d";
  return 0;
}
```

Result

```
 1. 31 11111
 2. 47 101111
 3. 55 110111
 4. 59 111011
 5. 61 111101
 6. 62 111110
 7. 79 1001111
 8. 87 1010111
 9. 91 1011011
 10. 93 1011101
 11. 94 1011110
 12. 103 1100111
 13. 107 1101011
 14. 109 1101101
 15. 110 1101110
 16. 115 1110011
 17. 117 1110101
 18. 118 1110110
 19. 121 1111001
 20. 122 1111010
 21. 124 1111100
 22. 143 10001111
 23. 151 10010111
 24. 155 10011011
 25. 157 10011101
 26. 158 10011110
 27. 167 10100111
 28. 171 10101011
```

```
29. 173 10101101
30. 174 10101110
31. 179 10110011
32. 181 10110101
33. 182 10110110
34. 185 10111001
35. 186 10111010
36. 188 10111100
37. 199 11000111
38. 203 11001011
39. 205 11001101
40. 206 11001110
41. 211 11010011
42. 213 11010101
43. 214 11010110
44. 217 11011001
45. 218 11011010
46. 220 11011100
47. 227 11100011
48. 229 11100101
49. 230 11100110
50. 233 11101001
51. 234 11101010
52. 236 11101100
53. 241 11110001
54. 242 11110010
55. 244 11110100
56. 248 11111000
57. 271 100001111
58. 279 100010111
59. 283 100011011
60. 285 100011101
61. 286 100011110
62. 295 100100111
63. 299 100101011
64. 301 100101101
65. 302 100101110
66. 307 100110011
67. 309 100110101
68. 310 100110110
69. 313 100111001
70. 314 100111010
71. 316 100111100
72. 327 101000111
73. 331 101001011
74. 333 101001101
75. 334 101001110
76. 339 101010011
77. 341 101010101
78. 342 101010110
79. 345 101011001
80. 346 101011010
81. 348 101011100
82. 355 101100011
83. 357 101100101
84. 358 101100110
85. 361 101101001
86. 362 101101010
```

```
87. 364 101101100
88. 369 101110001
89. 370 101110010
90. 372 101110100
91. 376 101111000
92. 391 110000111
93. 395 110001011
94. 397 110001101
95. 398 110001110
96. 403 110010011
97. 405 110010101
98. 406 110010110
99. 409 110011001
100. 410 110011010
101. 412 110011100
102. 419 110100011
103. 421 110100101
104. 422 110100110
105. 425 110101001
106. 426 110101010
107. 428 110101100
108. 433 110110001
109. 434 110110010
110. 436 110110100
111. 440 110111000
112. 451 111000011
113. 453 111000101
114. 454 111000110
115. 457 111001001
116. 458 111001010
117. 460 111001100
118. 465 111010001
119. 466 111010010
120. 468 111010100
121. 472 111011000
122. 481 111100001
123. 482 111100010
124. 484 111100100
125. 488 111101000
126. 496 111110000
Total 126
T h e   E n d
```

### Ghi nhớ một vài công thức tính nhanh

- $2^k = 1 << k$
- $x * 2^k = x << k$
- $x / 2^k = x \& 2^k - 1$

### Phương án 2. Tính nhẩm

Số $2^9$ trong hệ nhị phân có dạng 10 bit: $1000000000_2$. Ta cần chọn ra các số có 5 bit 1 trong số đó bit đầu tiên phải là 1. Vậy đáp số là

(tổ hợp chặp 5 của 10) / 2 = (tổ hợp chặp 4 của 9)

C(10, 5) / 2 = C(4,9)

$$C(n,k)/2 = \frac{n!}{2 \times k! \times (n-k)!} = \frac{2 \times 3 \times 4 \times 5 \times 6 \times 7 \times 8 \times 9 \times 10}{2 \times (2 \times 3 \times 4 \times 5) \times (2 \times 3 \times 4 \times 5)} =$$

$$= \frac{6 \times 7 \times 8 \times 9 \times 10}{2 \times 2 \times 3 \times 4 \times 5} = \frac{\cancel{6} \times 7 \times 2 \times 9 \times \cancel{10}}{\cancel{2} \times \cancel{2} \times \cancel{3} \times \cancel{4} \times \cancel{5}} = 7 \times 2 \times 9 = 126$$

## Problem 4. Vài hàm đệ quy

Để xây dựng một hàm đệ quy ta cần chú ý hai điểm quan trọng:

- Xây dựng hệ thức đệ quy R(x) = R(y)
- Xác định điều kiện kết thúc

### Độ cao của một số

```
int RH(Long x, int base = 10) {
  return (x == 0) ? 0 : RH(x / base, base) + (x % base);
}
```

- Hệ thức đệ quy RH(x, base) = RH(x / base) + (x % base)

Độ cao của số x = Độ cao của phần còn lại của x + chữ số đơn vị của x

- Xác định điều kiện kết thúc: x = 0

### Số chữ số của số nguyên x trong hệ đếm base: Len(x, base), RLen(x, base)

### Phương án không đệ quy: Len(x, base)

```
int Len(Long x, int base = 10) {
  int c = 0;
  while (x) {
     ++c;
     x /= base;
  }
  return c;
}
```

### Phương án đệ quy: RLen(x, base)

```
int RLen(Long x, int base = 10) {
  return (x < base) ? 1 : RLen(x / base, base) + 1;
}
```

- Hệ thức đệ quy `RLen(x, base) = RLen(x / base) + 1`

  Len của số x = Len(Phần còn lại của x) + chữ số đơn vị của x

- Xác định điều kiện kết thúc: nếu x < base thì x có 1 chữ số.

### Program

Xác định Len của các số trong hệ đếm 10 và hệ đếm 2.

```
#include <iostream>

using namespace std;

typedef long long Long;

const string DIGIT = "0123456789";
const string alpha = "abcdefghijklmnopqrstuvwxyz";
const string EDIGIT = DIGIT + alpha; // Expanded Digit

void Go() {
   cout << " ? ";
   fflush(stdin);
   if (cin.get() == '.') exit(0);
}

// -1234 -> "-1234"
string IntToStr(Long x, int base = 10) {
  string s = "";
  if (x == 0) return "0";
  string sign = "";
  if (x < 0) {
     sign = "-";
     x = -x;
 }
  while(x) {
    s = EDIGIT[x % base] + s;
    x /= base;
  }
  return sign + s;
}

int Len(Long x, int base = 10) {
  int c = 0;
  while (x) {
     ++c;
     x /= base;
  }
  return c;
}

int RLen(Long x, int base = 10) {
  return (x < base) ? 1 : RLen(x / base, base) + 1;
}

int Run1() {
  cout << "\n He dem 2: ";
  for (int x = 10; x < 10000; x = x * 2 + 1) {
     cout << "\n x = " << x << " Binary: " << IntToStr(x,2)
              << " Bit Len = " << RLen(x,2);
  }
  Go();
 for (int x = 0; x < 20; ++x) {
     cout << "\n x = " << x << "  Binary: " << IntToStr(x,2)
              << " " << " Bit Len = " << RLen(x,2);
  }
  Go();
}
```

```
int Run2() {
  cout << "\n He dem 10: ";
  for (int x = 0; x < 10000; x = x * 2 + 1) {
     cout << "\n x = " << x << " Decimal: "
              << " Bit Len = " << RLen(x);
  }
}

main() {
  Run1();
  Run2();
  cout << "\n T h e   E n d";
  return 0;
}
```

## Result

```
 He dem 2:
 x = 10 Binary: 1010 Bit Len = 4
 x = 21 Binary: 10101 Bit Len = 5
 x = 43 Binary: 101011 Bit Len = 6
 x = 87 Binary: 1010111 Bit Len = 7
 x = 175 Binary: 10101111 Bit Len = 8
 x = 351 Binary: 101011111 Bit Len = 9
 x = 703 Binary: 1010111111 Bit Len = 10
 x = 1407 Binary: 10101111111 Bit Len = 11
 x = 2815 Binary: 101011111111 Bit Len = 12
 x = 5631 Binary: 1010111111111 Bit Len = 13 ?

 x = 0  Binary: 0  Bit Len = 1
 x = 1  Binary: 1  Bit Len = 1
 x = 2  Binary: 10  Bit Len = 2
 x = 3  Binary: 11  Bit Len = 2
 x = 4  Binary: 100  Bit Len = 3
 x = 5  Binary: 101  Bit Len = 3
 x = 6  Binary: 110  Bit Len = 3
 x = 7  Binary: 111  Bit Len = 3
 x = 8  Binary: 1000  Bit Len = 4
 x = 9  Binary: 1001  Bit Len = 4
 x = 10  Binary: 1010  Bit Len = 4
 x = 11  Binary: 1011  Bit Len = 4
 x = 12  Binary: 1100  Bit Len = 4
 x = 13  Binary: 1101  Bit Len = 4
 x = 14  Binary: 1110  Bit Len = 4
 x = 15  Binary: 1111  Bit Len = 4
 x = 16  Binary: 10000  Bit Len = 5
 x = 17  Binary: 10001  Bit Len = 5
 x = 18  Binary: 10010  Bit Len = 5
 x = 19  Binary: 10011  Bit Len = 5 ?

 He dem 10:
 x = 0 Decimal:  Bit Len = 1
 x = 1 Decimal:  Bit Len = 1
 x = 3 Decimal:  Bit Len = 1
 x = 7 Decimal:  Bit Len = 1
 x = 15 Decimal:  Bit Len = 2
 x = 31 Decimal:  Bit Len = 2
```

```
x = 63 Decimal:  Bit Len = 2
x = 127 Decimal:  Bit Len = 3
x = 255 Decimal:  Bit Len = 3
x = 511 Decimal:  Bit Len = 3
x = 1023 Decimal:  Bit Len = 4
x = 2047 Decimal:  Bit Len = 4
x = 4095 Decimal:  Bit Len = 4
x = 8191 Decimal:  Bit Len = 4
T h e   E n d
```

## Tổ hợp chặp k của n phần tử: C(n,k)

### Formula

$$C(n,k) = \binom{n}{k} = \frac{n!}{k!\,(n-k)!} = \frac{n(n-1)\cdots(n-k+1)}{k(k-1)\cdots 1}$$

0! = 1.

Ta có

$$\binom{n}{k} = \frac{n!}{k!\,(n-k)!} = \frac{n!}{(k-1)!\,(n-k+1)!} \times \frac{n-k+1}{k} = \binom{n}{k-1} \times \frac{n-k+1}{k}$$

Nếu sử dụng cách ghi tuyến tính *C*(*n, k*) cho hàm tính tổ hợp chập *k* của *n,* ta có:

$$C(n, k) = C(n, k-1)(n-k+1) \text{ div } k \quad (1)$$

$$C(n, 0) = 1.$$

$$C(n, 1) = n.$$

Một phương án đệ quy cho hàm *C*(*n, k*) là:

C(n, k) = (k == 1) ? n : C(n, k – 1) * (n – k + 1) div k.

Thay *k* trong (1) bằng *i* + 1 ta thu được

$$C(n, i+1) = C(n, i)(n-i) \text{ div } (i+1) \quad (2)$$

Trong dạng không đệ quy, vận dụng (2), hàm *C*(*n, k*) sẽ được tính như sau:

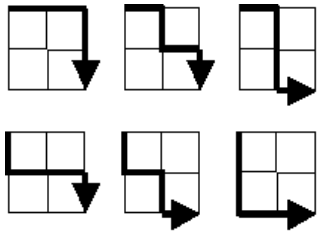
```
// non recursive of C(n, k), n > 0, k > 0
Long C(int n, int k) {
  Long r = 1;
  for (int i = 0; i < k; ++i) {
    r = r * (n-i) / (i+1);
  }
  return r;
}

// Recursive of C(n, k), n > 0, k > 0
Long RC(int n, int k) {
  return (k == 1)? n : RC(n,k-1)*(n-k+1)/k;
}
```

**Commented [WU1]:** Việt hóa dòng này

**Commented [WU2]:** Việt hóa dòng này

### Ví dụ 1 (Problem 15, Project Euler)

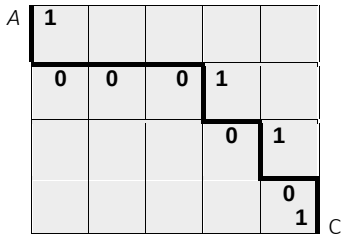
Xuất phát từ góc trên trái của lưới kích thước 2×2 có 6 đường (không quay lui) dẫn đến góc dưới phải như sau:



Có bao nhiêu đường như vậy trong lưới 20×20?

### Algorithm

Ta gắng giải bài này với lưới tổng quát có kích thước *a*×*b*, trong đó *a* là chiều rộng, *b* là chiều dài của lưới. Trước hết ta gọi một đường không quay lui từ *A* đến *C* là *đường hợp lệ*. Trong mỗi đường, ta gán nhãn 0 cho cạnh đơn vị ngang và 1 cho cạnh dọc.




*Đường 100010101 trong lưới 4×5. a = 4, b = 5.*

Ta thấy mọi đường hợp lệ đều đi qua đúng *a* cạnh dọc và đúng *b* cạnh ngang. Từ đây suy ra mọi đường hợp lệ đều có chiều dài *a*+*b*. Vậy bài toán có thể được phát biểu lại như sau:

*Có bao nhiêu xâu nhị phân dài a+b chứa đúng a số* 1 (hoặc *b* số 0).

Đáp số khi đó sẽ là $\binom{n}{k}$ - *tổ hợp chập k của n* với công thức tính

### Program

```
#include <iostream>

using namespace std;

#include <iostream>
```

```
using namespace std;

typedef long long Long;

// non recursive of C(n, k)
Long C(int n, int k) {
  Long r = 1;
  for (int i = 0; i < k; ++i) {
    r = r * (n-i) / (i+1);
  }
  return r;
}

// Recursive of C(n, k)
Long RC(int n, int k) {
  return (k == 1)? n : RC(n,k-1)*(n-k+1)/k;
}

main() {
  Long n = 40, k = 20;
  cout << C(n, k) << " " << RC(n,k); // 137846528820
  cout << "\n T h e   E n d";
  return 0;
}
```

### Ví dụ 2

Trong các số nhị phân từ 1 đến $2^9$ có bao nhiêu số chứa đúng 5 bit 1 ?

Đáp số

```
C(9, 4) or RC(9, 4)
```

### Some functions on strings

Tạo một khúc string từ s[i] đến s[j] tứ string s.

```
string Segment(string s, int i, int j = 10000000) {
  string w = "";
  if (i < 0) i = 0;
  if (j >= s.length()) j = s.length()-1;
  for (int k = i; k <= j; ++k) {
      w += s[k];
 }
 return w;
}
```

Lật string s cho ra một string mới, s không thay đổi.

```
string Rev(string & s) {
  string w = "";
  for (int i = s.length() - 1; i >= 0; --i) {
    w += s[i];
 }
```

```
 return w;
}
```

Lật string s cho ra một string mới, s không thay đổi. Phương án đệ quy.

```
string RRev(string s) {
  return (s == "") ? s : RRev(Segment(s,1)) + s[0];
}
```

Cho ra string mới trong đó chuyển mọi ký tự trong s thành chữ hoa, s không thay đổi.

```
string ToUpper(string &s) {
  string w = "";
  for (int i = 0; i < s.length(); ++i) {
      w += toupper(s[i]);
 }
 return w;
}
```

Cho ra string mới trong đó chuyển mọi ký tự trong s thành chữ thường, s không thay đổi.

```
string ToLower(string &s) {
  string w = "";
  for (int i = 0; i < s.length(); ++i) {
      w += tolower(s[i]);
 }
 return w;
}
```

Tạo string mới trong đó nọi ký tự c1 của s được thay bằng c2, s không thay đổi.

Nếu c2 = 0 có nghĩa là xóa mọi ký tự c1 trong s.

```
// c2 = 0 means delete all c1
string Replace(string &s, char c1, char c2 = 0) {
  string w = "";
  if (c2 == 0) {
     for (int i = 0; i < s.length(); ++i) {
         if (s[i] != c1) w += s[i];
     }
   return w;
 }
 for (int i = 0; i < s.length(); ++i) {
    w += (s[i] == c1) ? c2 : s[i];
 }
 return w;
}
```

Program

```
#include <iostream>

using namespace std;

void Go() {
   cout << " ? ";
   fflush(stdin);
   if (cin.get() == '.') exit(0);
}

string Segment(string s, int i, int j = 10000000) {
  string w = "";
  if (i < 0) i = 0;
  if (j >= s.length()) j = s.length()-1;
  for (int k = i; k <= j; ++k) {
      w += s[k];
 }
 return w;
}

string Rev(string & s) {
  string w = "";
  for (int i = s.length() - 1; i >= 0; --i) {
    w += s[i];
 }
 return w;
}

string RRev(string s) {
  return (s == "") ? s : RRev(Segment(s,1)) + s[0];
}

string ToUpper(string &s) {
  string w = "";
  for (int i = 0; i < s.length(); ++i) {
      w += toupper(s[i]);
 }
 return w;
}

string ToLower(string &s) {
  string w = "";
  for (int i = 0; i < s.length(); ++i) {
      w += tolower(s[i]);
 }
 return w;
}

// c2 = 0 means delete all c1
string Replace(string &s, char c1, char c2 = 0) {
  string w = "";
  if (c2 == 0) {
     for (int i = 0; i < s.length(); ++i) {
         if (s[i] != c1) w += s[i];
     }
```

```
   return w;
 }
 for (int i = 0; i < s.length(); ++i) {
    w += (s[i] == c1) ? c2 : s[i];
 }
 return w;
}

void Test() {
   string s = "abba and Panama";
   cout << "\n Phase 1: Givaen " << s;
   for (int i = 0; i < s.length(); ++i)
     cout << "\n " << Segment(s, i);
   cout << "\n RRev: |" << RRev(s) << "|";
   s = ToUpper(s);
   cout << "\n " << "|" << s << "| -> |" << ToLower(s) << "|";
   cout << "\n Phase 2: Given " << "|" << s << "|";
   s = Replace(s,'A','*');
   cout << "\n Replace A by *: " << "|" << s << "|";
   s = "    " + s + "     ";
   cout << "\n Add spaces: |" << s << "|";
   s = Replace(s,' '); // or 32
   cout << "\n del all spaces: |" << s << "|";
   s = Replace(s,'*');
   cout << "\n del all *: |" << s << "|";
}

main() {
  Test();
  cout << "\n T h e   E n d";
  return 0;
}
```

## Result

```
Phase 1: Givaen abba and Panama
abba and Panama
bba and Panama
ba and Panama
a and Panama
 and Panama
and Panama
nd Panama
d Panama
 Panama
Panama
anama
nama
ama
ma
a
RRev: |amanaP dna abba|
|ABBA AND PANAMA| -> |abba and panama|
Phase 2: Given |ABBA AND PANAMA|
Replace A by *: |*BB* *ND P*N*M*|
Add spaces: |    *BB* *ND P*N*M*     |
```

```
del all spaces: |*BB**NDP*N*M*|
del all *: |BBNDPNM|
T h e   E n d
```