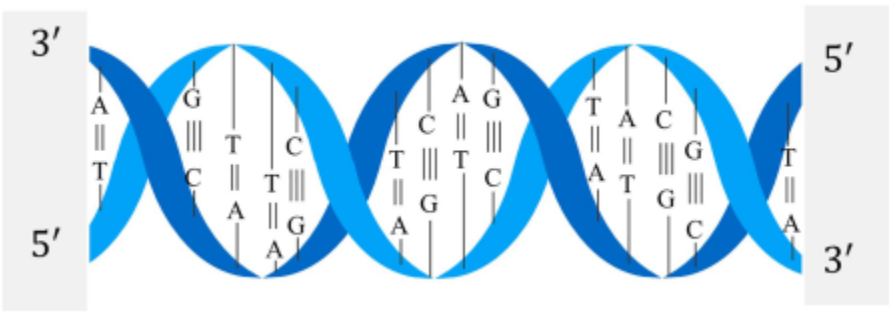

DNA Bank

[문제] 어떤 유전자 은행은 생물학적 무기로 사용할 수 있는 세균의 DNA를 관리하고 있다. 이 DNA는 N개의 segment S_i로 관리된다. 우리는 이 DNA S_i를 사전식으로 정렬할 때 중간 몇 개의 S_i를 찾으려고 한다. 그런데 이 유전자 은행에서는 보안을 위해서 S_i 서열 전체를 공개하지는 않고 연구자들이 요청한 특정 S_i 서열의 j번째 뉴클레오타이드(nucleotide)만을 알려준다. 이 작업은 다음의 함수로 확인할 수 있다.

```
DNA_base( seed, i, j )
```

이 함수는 S_i segment의 j번째 뉴클레오타이드(nucleotide) 문자를 `{a,c,t,g,$}`에서 골라 돌려준다. 만일 j가 S_i의 길이를 넘을 경우에는 문자 '`$`'가 return된다. 즉 $|S_i| = 1234$ 일단 $j = 1256$ 이면 위 함수의 결과는 '`$`'가 된다. 이 문제에서는 입력 파일은 따로 제공하지 않으며 위 함수만을 이용해서 풀어야 한다.

여러분은 N개의 DNA segment를 사전식(lexicographical)으로 정렬했을 때 특정 번째 순서에 해당하는 DNA segment S_r의 index r를 찾아서 출력해야 한다.

[입출력] 이번 문제에서 입력 파일은 따로 없으며 head file에 `#include "DNA_bank.h"`를 선언하고, 제시된 함수를 사용하면 특정 segment의 j번째 뉴클레오타이드(nucleotide) 문자를 알아낼 수 있다. bank에 준비된 S_i의 개수 N의 범위는 $100 \le N \le 100{,}000$ 이며 S_i의 길이 범위는 $10 \le |S_i| \le 100$ 이다.

이 문제의 정답은 준비된 함수를 이용하여 보고해야 한다. 여러분은 프로그램을 통하여 찾아야 할 index 순서를 나타내는 정수 index1, index2를 받는다. 이것을 바탕으로 이 index1, index2에 해당하는 DNA sequence S_r의 index $r1, r2$를 찾아서 출력해야 한다. 단 $r1 < r2$ 이다. 이것을 준비된 output 함수 report(index1, index2)불러서 보고한다. 이 report( )함수를 부르면 전체 프로그램은 자동적으로 종료된다.

[제한조건] 프로그램 이름은 `DNABank.{cpp}`이며 제출 허용 횟수는 25회이다. 데이터 당 제한시간은 최대 1초, 그리고 token은 최대 700개이다. 단 이번 과제부터는 #define 문자열을 사용하여 코드의 길이를 줄이는 편법은 사용할 수 없다. #define은 반드시 1줄만 가능하며 multiline string은 더 이상 허용하지 않는다. 만일 이 방법을 사용하면 이전 점수에 관계없이 0점 처리된다. 이번 문제는 C++만 허용한다.

```
#include "DNA_bank.h"     // 강사가 준비합니다.

int main( ) {
   long long xseed, Nsize ;
   char nucleo ;
   int  index1, index2 ;
   int  fidx1, fidx2   ;
   set_DNA_bank( ) ;       // prepare data set !
   Nsize = get_size( ) ;   // get the number of segments
   xseed = get_seed( ) ;   // get seed number for work
   get_index( &index1, &index2 )
    ......
    nucleo = DNA_base ( xseed, i, j ) ;
    ...
    report( fidx1, fidx2 ) ; // report and quit
}
```

[참고자료]

```
#include <iostream>
#include <iomanip>
#include <cmath>

using namespace std;

char DNA_base(long long seed, int i, int j) {
    char Rbase[4]={'a', 'c', 'g', 't'} ;
    long long a = 1664525, c = 1013904223, m = pow(2, 32)-1;
    long long nseed  ;
    nseed = (a * seed + c) % m;
    int leng = nseed % 1000 + 10;

    if (j >= leng | j <= 0 ) return '$';

    for (int w = 0; w < j; w++) {
        nseed = (a * nseed + c) % m;
        //cout << "> " << nseed << ":  " << nseed%4 << endl ;
    }
    return Rbase[ nseed%4] ;
} // end of DNA_base( ) function

int main() {
    long long seed = 14223;
    char dnachar ;
    int lidx = 1202;


    for (int loop=1 ; loop <= 10 ; loop++ ) {
        cout << "\n" ; // seed= " << seed+loop << endl ;
        for (int w = 1; w < 100; w++) {
            dnachar = DNA_base(seed+loop, lidx, w);
            cout << dnachar ;
            if ( dnachar == '$') break;
        }
    } // end of loop
    return 0;
}  // end of main( )
```