Contents

Problem	Tags
01 A	
02 B	
03 C	
04 D	
05 E	
06 F	
07 G	
08 H	
09 I	
10 J	
11 K	
12 L	
13 M	

Time	Meeting Description	Chk
030	All Problems Read. Write Tags.	
060	Ace Decided. Choose Coder.	
090	Decide & Order Solveable Problems	
120	Status Check	
150	Status Check	
180	Status Check	
210	Status Check	
240	Status Check	
270	Status Check	

1 2D Geometry

1.1 Primitives

```
1  typedef complex<double> point;
2  struct circle {
3    point c; double r;
4    circle(point c, double r):c(c),r(r){}
5    circle(){}
6  };
7  double cross(const point &a, const point &b) {
8    return imag(conj(a)*b);
9  }
10 double dot(const point &a, const point &b) {
11   return real(conj(a)*b);
12 }
```

1.2 Intersections

```
1  // Line - Line
2  // Algorithm
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // Line - Segment
8  // Algorithm
9  // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Segment - Segment
14 // Algorithm
15 // Algorithm
16 // Algorithm
17 // Algorithm
18 // Algorithm
19 // Circle - Line
20 // Algorithm
21 // Algorithm
22 // Algorithm
23 // Algorithm
24 // Algorithm
25 // Circle - Segment
26 // Algorithm
27 // Algorithm
28 // Algorithm
29 // Algorithm
30 // Algorithm
31 // Circle - Circle
32 // Algorithm
33 // Algorithm
34 // Algorithm
35 // Algorithm
36 // Algorithm
37 // Line - Point
38 // Algorithm
39 // Algorithm
40 // Algorithm
41 // Algorithm
42 // Algorithm
43 // Segment - Point
44 // Algorithm
45 // Algorithm
46 // Algorithm
47 // Algorithm
48 // Algorithm
```

1.3 Circle Generation

```
1  // From 3 Points
2  // Algorithm
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // From 1 Line 2 Points
8  // Algorithm
9  // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // From 2 Lines 1 Point
14 // Algorithm
15 // Algorithm
16 // Algorithm
17 // Algorithm
18 // Algorithm
19 // From 3 Lines
20 // Algorithm
21 // Algorithm
22 // Algorithm
23 // Algorithm
24 // Algorithm
```

1.4 Heron Triangle Area

```
1  // Formula
2  // Formula
3  // Formula
```

1.5 Polygon Centroid

```
1  for(int i = 1; i < n-1; i++) {
2    pt   ai = pts[i] - pts[i-1],
3       ib = pts[i+1] - pts[i];
4    area += (conj(ai)*ib).imag();
5  }
```

1.6 Point In Polygon

```
1  // Algorithm
2  // Algorithm
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // Algorithm
8  // Algorithm
9  // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
16 // Algorithm
17 // Algorithm
18 // Algorithm
19 // Algorithm
20 // Algorithm
```

1.7   Convex Hull

```
1  // Algorithm
2  // Algorithm
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // Algorithm
8  // Algorithm
9  // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
16 // Algorithm
17 // Algorithm
18 // Algorithm
19 // Algorithm
20 // Algorithm
```

1.8   Line Segment Set Intersection

```
1  // Algorithm
2  // Algorithm
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // Algorithm
8  // Algorithm
9  // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
16 // Algorithm
17 // Algorithm
18 // Algorithm
19 // Algorithm
20 // Algorithm
21 // Algorithm
22 // Algorithm
23 // Algorithm
24 // Algorithm
25 // Algorithm
26 // Algorithm
27 // Algorithm
28 // Algorithm
29 // Algorithm
30 // Algorithm
31 // Algorithm
32 // Algorithm
33 // Algorithm
34 // Algorithm
35 // Algorithm
36 // Algorithm
37 // Algorithm
38 // Algorithm
39 // Algorithm
40 // Algorithm
41 // Algorithm
42 // Algorithm
43 // Algorithm
44 // Algorithm
45 // Algorithm
46 // Algorithm
47 // Algorithm
48 // Algorithm
49 // Algorithm
```

1.9   Voronoi Diagrams

```
1  // Algorithm
2  // Algorithm
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // Algorithm
8  // Algorithm
```

```
9  // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
16 // Algorithm
17 // Algorithm
18 // Algorithm
19 // Algorithm
20 // Algorithm
```

2   3D Geometry

2.1   Primitives

```
1  // Code
2  // Code
3  // Code
4  // Code
5  // Code
6  // Code
7  // Code
8  // Code
9  // Code
10 // Code
11 // Code
12 // Code
13 // Code
14 // Code
15 // Code
```

2.2   Convex Hull

```
1  // Algorithm
2  // Algorithm
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // Algorithm
8  // Algorithm
9  // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
16 // Algorithm
17 // Algorithm
18 // Algorithm
19 // Algorithm
20 // Algorithm
```

2.3   Great Circle Distance

```
1  // Code
2  // Code
3  // Code
4  // Code
5  // Code
6  // Code
7  // Code
8  // Code
9  // Code
10 // Code
```

3   Combinatorics

3.1   Basics

```
1  // catalan numbers
2  long long C(int n) {
3    return (C(n-1)*2*n*(2*n-1))/(n*(n+1));
4    return NCR(2*n, n) - NCR(2*n, n+1);
5    return NCR(2*n, n)/(n+1);
6  }
7  // derangements
8  long long D(int n) {
9    return n*D(n-1) + pow(-1, n);
10   return (n-1)*(D(n-1) + D(n-2));
11 }
12 // iterate over all subsets with < m elements
```

```
13  for (int i = 0; i < (1<<n); i=Integer.bitCount(i) < m ? i
        +1 : (i|(i-1))+1)
14  // iterate over all the subsets
15  for (int i=0; i < (1<<n); i++)
16    // iterate over all the subsets of the i-th subset
17    for(int i2 = i; i2 > 0; i2 = (i2-1) & i)
```

4 Data Structures

4.1 Palindromic Tree

```
1   // Algorithm
2   // Algorithm
3   // Algorithm
4   // Algorithm
5   // Algorithm
6   // Algorithm
7   // Algorithm
8   // Algorithm
9   // Algorithm
10  // Algorithm
11  // Algorithm
12  // Algorithm
13  // Algorithm
14  // Algorithm
15  // Algorithm
16  // Algorithm
17  // Algorithm
18  // Algorithm
19  // Algorithm
20  // Algorithm
```

4.2 Treap

```
1   // Algorithm
2   // Algorithm
3   // Algorithm
4   // Algorithm
5   // Algorithm
6   // Algorithm
7   // Algorithm
8   // Algorithm
9   // Algorithm
10  // Algorithm
11  // Algorithm
12  // Algorithm
13  // Algorithm
14  // Algorithm
15  // Algorithm
16  // Algorithm
17  // Algorithm
18  // Algorithm
19  // Algorithm
20  // Algorithm
21  // Algorithm
22  // Algorithm
23  // Algorithm
24  // Algorithm
25  // Algorithm
26  // Algorithm
27  // Algorithm
28  // Algorithm
29  // Algorithm
30  // Algorithm
```

4.3 Sparse Array

```
1   // Code
2   // Code
3   // Code
4   // Code
5   // Code
6   // Code
7   // Code
8   // Code
9   // Code
10  // Code
11  // Code
12  // Code
13  // Code
14  // Code
15  // Code
```

4.4 Skip Lists

```
1   // Code
2   // Code
3   // Code
4   // Code
5   // Code
6   // Code
7   // Code
8   // Code
9   // Code
10  // Code
11  // Code
12  // Code
13  // Code
14  // Code
15  // Code
16  // Code
17  // Code
18  // Code
19  // Code
20  // Code
21  // Code
22  // Code
```

4.5 BIT + Search

```
1   // Algorithm
2   // Algorithm
3   // Algorithm
4   // Algorithm
5   // Algorithm
6   // Algorithm
7   // Algorithm
8   // Algorithm
9   // Algorithm
10  // Algorithm
11  // Algorithm
12  // Algorithm
13  // Algorithm
14  // Algorithm
15  // Algorithm
16  // Algorithm
17  // Algorithm
18  // Algorithm
19  // Algorithm
20  // Algorithm
```

4.6 Segment Tree + Lazy Propagation

```
1   // Algorithm
2   // Algorithm
3   // Algorithm
4   // Algorithm
5   // Algorithm
6   // Algorithm
7   // Algorithm
8   // Algorithm
9   // Algorithm
10  // Algorithm
11  // Algorithm
12  // Algorithm
13  // Algorithm
14  // Algorithm
15  // Algorithm
16  // Algorithm
17  // Algorithm
18  // Algorithm
19  // Algorithm
20  // Algorithm
21  // Algorithm
22  // Algorithm
23  // Algorithm
24  // Algorithm
25  // Algorithm
26  // Algorithm
27  // Algorithm
28  // Algorithm
29  // Algorithm
30  // Algorithm
```

4.7 Weighted Union Disjoint Sets

```
1   // Algorithm
2   // Algorithm
3   // Algorithm
4   // Algorithm
```

```
5  // Algorithm
6  // Algorithm
7  // Algorithm
8  // Algorithm
9  // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
```

5   Game Theory

5.1   Nim Game

```
1  // Algorithm
2  // Algorithm
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // Algorithm
8  // Algorithm
9  // Algorithm
10 // Algorithm
```

5.2   Grundy Numbers

```
1  // Algorithm
2  // Algorithm
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // Algorithm
8  // Algorithm
9  // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
16 // Algorithm
17 // Algorithm
18 // Algorithm
19 // Algorithm
20 // Algorithm
```

5.3   General Josephus Problem

6   General Mathematics

6.1   Inclusion-Exclusion Patterns

```
1  // Algorithm
2  // Algorithm
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // Algorithm
8  // Algorithm
9  // Algorithm
10 // Algorithm
```

6.2   Determinant

```
1  // Algorithm
2  // Algorithm
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // Algorithm
8  // Algorithm
9  // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
```

6.3   Gaussian Elimination

```
1  double* GaussianElimination(int N, double **mat) {
2    int i, j, k, L; double t;
3    for (i = 0; i < N - 1; i++) {
4      L = i;
5      for (j = i + 1; j < N; j++)
6        if (fabs(mat[j][i]) > fabs(mat[L][i]))
7          L = j;
8      for (k = i; k <= N; k++)
9      swap(mat[i][k], mat[L][k]);
10     for (j = i + 1; j < N; j++)
11       for (k = N; k >= i; k--)
12         mat[j][k] -= (mat[i][k] * mat[j][i]) / mat[i][i];
13   }
14   double *res = new double[N];
15   for (j = N - 1; j >= 0; j--) {
16     for (t = 0.0, k = j + 1; k < N; k++)
17     t += mat[j][k] * res[k];
18     res[j] = (mat[j][N] - t) / mat[j][j];
19   }
20   return res;
21 }
```

6.4   Fast Fourier-Transform

```
1  // Algorithm
2  // Algorithm
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // Algorithm
8  // Algorithm
9  // Algorithm
10 // Algorithm
```

6.5   Misc. Formulas

```
1  // Algorithm
2  // Algorithm
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // Algorithm
8  // Algorithm
9  // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
16 // Algorithm
17 // Algorithm
18 // Algorithm
19 // Algorithm
20 // Algorithm
21 // Algorithm
22 // Algorithm
23 // Algorithm
24 // Algorithm
25 // Algorithm
26 // Algorithm
27 // Algorithm
28 // Algorithm
29 // Algorithm
30 // Algorithm
```

7   Graph Theory

7.1   Primitives

```
1  // Algorithm
2  // Algorithm
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // Algorithm
8  // Algorithm
9  // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
```

```
14 // Algorithm
15 // Algorithm
```

7.2 Articulation Points & Bridges

```
 1 // Algorithm
 2 // Algorithm
 3 // Algorithm
 4 // Algorithm
 5 // Algorithm
 6 // Algorithm
 7 // Algorithm
 8 // Algorithm
 9 // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
16 // Algorithm
17 // Algorithm
18 // Algorithm
19 // Algorithm
20 // Algorithm
```

7.3 SCC

```
 1 // Algorithm
 2 // Algorithm
 3 // Algorithm
 4 // Algorithm
 5 // Algorithm
 6 // Algorithm
 7 // Algorithm
 8 // Algorithm
 9 // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
16 // Algorithm
17 // Algorithm
18 // Algorithm
19 // Algorithm
20 // Algorithm
```

7.4 2-SAT

```
 1 // Algorithm
 2 // Algorithm
 3 // Algorithm
 4 // Algorithm
 5 // Algorithm
 6 // Algorithm
 7 // Algorithm
 8 // Algorithm
 9 // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
16 // Algorithm
17 // Algorithm
18 // Algorithm
19 // Algorithm
20 // Algorithm
```

7.5 Edmonds-Karp Max Flow

```
 1 // Algorithm
 2 // Algorithm
 3 // Algorithm
 4 // Algorithm
 5 // Algorithm
 6 // Algorithm
 7 // Algorithm
 8 // Algorithm
 9 // Algorithm
10 // Algorithm
```

```
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
16 // Algorithm
17 // Algorithm
18 // Algorithm
19 // Algorithm
20 // Algorithm
```

7.6 Dinic's Max Flow

```
 1 // Algorithm
 2 // Algorithm
 3 // Algorithm
 4 // Algorithm
 5 // Algorithm
 6 // Algorithm
 7 // Algorithm
 8 // Algorithm
 9 // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
16 // Algorithm
17 // Algorithm
18 // Algorithm
19 // Algorithm
20 // Algorithm
```

7.7 Min-Cost Max Flow

```
 1 // Algorithm
 2 // Algorithm
 3 // Algorithm
 4 // Algorithm
 5 // Algorithm
 6 // Algorithm
 7 // Algorithm
 8 // Algorithm
 9 // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
16 // Algorithm
17 // Algorithm
18 // Algorithm
19 // Algorithm
20 // Algorithm
```

7.8 Euler Cycles

```
 1 // Algorithm
 2 // Algorithm
 3 // Algorithm
 4 // Algorithm
 5 // Algorithm
 6 // Algorithm
 7 // Algorithm
 8 // Algorithm
 9 // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
16 // Algorithm
17 // Algorithm
18 // Algorithm
19 // Algorithm
20 // Algorithm
```

7.9 Maximum Matching

```
 1 // Algorithm
 2 // Algorithm
```

```
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // Algorithm
8  // Algorithm
9  // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
16 // Algorithm
17 // Algorithm
18 // Algorithm
19 // Algorithm
20 // Algorithm
```

7.10 HL Decomposition

```
1  // Algorithm
2  // Algorithm
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // Algorithm
8  // Algorithm
9  // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
16 // Algorithm
17 // Algorithm
18 // Algorithm
19 // Algorithm
20 // Algorithm
```

7.11 Modelling Inequalities

```
1  // Algorithm
2  // Algorithm
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // Algorithm
8  // Algorithm
9  // Algorithm
10 // Algorithm
```

7.12 Max Flow Tricks

```
1  // Algorithm
2  // Algorithm
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // Algorithm
8  // Algorithm
9  // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
```

7.13 Bellman Ford

```
1  // Algorithm
2  // Algorithm
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // Algorithm
8  // Algorithm
9  // Algorithm
10 // Algorithm
```

7.14 Stable Marriage

```
1  // Algorithm
2  // Algorithm
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // Algorithm
8  // Algorithm
9  // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
```

7.15 Maximum Assignment

```
1  // Algorithm
2  // Algorithm
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // Algorithm
8  // Algorithm
9  // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
```

8 Linear Programming

8.1 Simplex

```
1  // Algorithm
2  // Algorithm
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // Algorithm
8  // Algorithm
9  // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
16 // Algorithm
17 // Algorithm
18 // Algorithm
19 // Algorithm
20 // Algorithm
21 // Algorithm
22 // Algorithm
23 // Algorithm
24 // Algorithm
25 // Algorithm
26 // Algorithm
27 // Algorithm
28 // Algorithm
29 // Algorithm
30 // Algorithm
31 // Algorithm
```

9 Number Theory

9.1 Extended GCD

```
1  long long gcd( long long a, long long b )
2  { return( b == 0 ? a : gcd( b, a % b ) ); }
3  //USED BY: egcd, msolve, inverse, ldioph
4  template< class Int > struct Triple {
5    Int d, x, y;
6    Triple(Int q, Int w, Int e):d(q), x(w), y(e){}
7  };
8  //USED BY: msolve, inverse, ldioph
9  template< class Int > Triple< Int > egcd( Int a, Int b ) {
```

```
10    if( !b ) return Triple< Int >( a, Int( 1 ), Int( 0 ) );
11    Triple< Int > q = egcd( b, a % b );
12    return Triple< Int >( q.d, q.y, q.x - a / b * q.y );
13 }
```

9.2 Modular Inverse

```
1 //solves ax = 1 (mod n).
2 template< class Int > Int inverse( Int a, Int n ) {
3     Triple< Int > t = egcd( a, n );
4     if( t.d > Int( 1 ) ) return Int( 0 );
5     Int r = t.x % n;
6     return( r < Int( 0 ) ? r + n : r );
7 }
```

9.3 Modular Linear Equation

```
1 // Algorithm
2 // Algorithm
3 // Algorithm
4 // Algorithm
5 // Algorithm
6 // Algorithm
7 // Algorithm
8 // Algorithm
9 // Algorithm
10 // Algorithm
```

9.4 Linear Diophantine Equation

```
1 // Algorithm
2 // Algorithm
3 // Algorithm
4 // Algorithm
5 // Algorithm
6 // Algorithm
7 // Algorithm
8 // Algorithm
9 // Algorithm
10 // Algorithm
```

9.5 Modular Powers

```
1 // Algorithm
2 // Algorithm
3 // Algorithm
4 // Algorithm
5 // Algorithm
6 // Algorithm
7 // Algorithm
8 // Algorithm
9 // Algorithm
10 // Algorithm
```

9.6 Sieve of Eratosthenes

```
1 // Algorithm
2 // Algorithm
3 // Algorithm
4 // Algorithm
5 // Algorithm
6 // Algorithm
7 // Algorithm
8 // Algorithm
9 // Algorithm
10 // Algorithm
```

9.7 Primality Testing & Factoring

```
1 // Algorithm
2 // Algorithm
3 // Algorithm
4 // Algorithm
5 // Algorithm
6 // Algorithm
7 // Algorithm
8 // Algorithm
9 // Algorithm
10 // Algorithm
```

9.8 Euler Phi

```
1 // Algorithm
2 // Algorithm
3 // Algorithm
4 // Algorithm
5 // Algorithm
6 // Algorithm
7 // Algorithm
8 // Algorithm
9 // Algorithm
10 // Algorithm
```

9.9 Chinese Remainder

```
1 // Algorithm
2 // Algorithm
3 // Algorithm
4 // Algorithm
5 // Algorithm
6 // Algorithm
7 // Algorithm
8 // Algorithm
9 // Algorithm
10 // Algorithm
```

9.10 Discrete Logarithm

```
1 // Algorithm
2 // Algorithm
3 // Algorithm
4 // Algorithm
5 // Algorithm
6 // Algorithm
7 // Algorithm
8 // Algorithm
9 // Algorithm
10 // Algorithm
```

9.11 Tortoise & Hare

```
1  // mu = start of cycle, lambda = cycle length
2  ii floyd(int x0) {
3    int tortoise = f(x0), hare = f(f(x0));
4    while(tortoise != hare)
5      tortoise = f(tortoise), hare = f(f(hare));
6    int mu = 0; hare = x0;
7    while(tortoise != hare)
8      tortoise = f(tortoise), hare = f(hare), mu++;
9    int lambda = 1; hare = f(tortoise);
10   while(tortoise != hare)
11     hare = f(hare), lambda++;
12   return ii(mu, lambda);
13 }
```

9.12 Pollard Rho
10 Search
10.1 Binary Search

```
1  // Algorithm
2  // Algorithm
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // Algorithm
8  // Algorithm
9  // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
```

10.2 Ternary Search

```
1 long double min() {
2     long double lo = -1e6, hi = 1e6, res = 3e6;
3     while(fabs(lo-hi) > EPS) {
4         long double left = (hi-lo)/3 + lo, right = (2*(hi-
              lo))/3 + lo;
5         long double resL = F(left), resR = F(right);
6         if(resL < resR)
7             hi = right;
```

```
8          else
9              lo = left;
10          res = min(res, min(resL, resR));
11      }
12      return res;
13 }
```

11　Strings

11.1　Aho Corasick

```
1  // Algorithm
2  // Algorithm
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // Algorithm
8  // Algorithm
9  // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
```

11.2　Hashing

```
1  // Algorithm
2  // Algorithm
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // Algorithm
8  // Algorithm
9  // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
```

11.3　Z-Algorithm

```
1  // Algorithm
2  // Algorithm
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // Algorithm
8  // Algorithm
9  // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
```

11.4　KMP + Periods

```
1  // Algorithm
2  // Algorithm
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // Algorithm
8  // Algorithm
9  // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
```

11.5　Manacher

```
1  // Algorithm
2  // Algorithm
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // Algorithm
8  // Algorithm
9  // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
16 // Algorithm
17 // Algorithm
18 // Algorithm
19 // Algorithm
20 // Algorithm
21 // Algorithm
```

11.6　Suffix Array

```
1  // Algorithm
2  // Algorithm
3  // Algorithm
4  // Algorithm
5  // Algorithm
6  // Algorithm
7  // Algorithm
8  // Algorithm
9  // Algorithm
10 // Algorithm
11 // Algorithm
12 // Algorithm
13 // Algorithm
14 // Algorithm
15 // Algorithm
16 // Algorithm
17 // Algorithm
18 // Algorithm
19 // Algorithm
20 // Algorithm
21 // Algorithm
22 // Algorithm
23 // Algorithm
24 // Algorithm
25 // Algorithm
26 // Algorithm
27 // Algorithm
28 // Algorithm
29 // Algorithm
30 // Algorithm
```

12　Misc