26ος ΠΑΝΕΛΛΗΝΙΟΣ ΔΙΑΓΩΝΙΣΜΟΣ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΘΕΜΑΤΑ ΤΕΛΙΚΗΣ ΦΑΣΗΣ

Θέμα 1°: Άθροισμα ζευγών [30 Μονάδες]

C++

Αριστοφάνης Ροντογιάννης

```
#include <stdio.h>
#include <algorithm>

using namespace std;

const int MAXN = 1000002;
int a[MAXN];

int main() {
#ifdef CONTEST
  freopen("sumpair.in", "r", stdin);
  freopen("sumpair.out", "w", stdout);
#else
  freopen("input.txt", "r", stdin);
  freopen("output2.txt", "w", stdout);
#endif
  int n, q;
  scanf("%d %d", &n, &q);
  for(int i=0; i<n; ++i)
    scanf("%d", a+i);
  sort(a, a+n);
  while(q--) {
    int val;
    scanf("%d", &val);
    int ok = 0;
    int lo = 0, hi = n-1;
    while(lo < hi) {
      if(a[lo] + a[hi] > val) hi--;
      else if(a[lo] + a[hi] < val) lo++;
      else {
        ok = 1;
        break;
      }
    }
    if(ok) printf("true\n");
    else printf("false\n");
  }
  return 0;
}
```

Θέμα 2ο: Επισκευή δρόμου [35 Μονάδες]

C++

Χρήστος Πόριος (92%)

```
#include <cstdio>

inline int max(int a, int b) {
    return (a>b)?a:b;
}

struct Node {
    int lo, hi, mid, range;
    int rval, lval, mval, val;
    bool painted;
    Node *left, *right;

    void make_children() {
        if(range==1) return;
        if(left == 0) left = new Node(lo, mid, painted);
        if(right == 0) right = new Node(mid+1, hi, painted);
    }

    void update() {
      if(range==1) return;
      lval = left->lval;
      if(left->val == left->range) lval = left->val + right-
>lval;

      rval = right->rval;
      if(right->val == right->range) rval = right->val + left-
>rval;

      mval = max(max(left->val, right->val), left->rval +
right->lval);

      val = max(max(lval, rval), mval);
    }

    void paint(int qlo, int qhi) {
        if(qlo <= lo && qhi >= hi) {
            rval = lval = mval = val = 0;
            painted = true;
            return;
        }

        make_children();
        if(qlo <= mid) left->paint(qlo, qhi);
        if(qhi > mid) right->paint(qlo, qhi);
        update();
```

```
    }

    Node(int l, int h, bool p) {
        //printf("Building node for [%d, %d] (%c)\n", l, h,
painted?'B':'W');
        lo = l; hi = h;
        mid = (lo+hi)/2;
        range = hi-lo+1;
        painted = p;

        left = right = 0;

        if(painted) rval = lval = mval = val = 0;
        else rval = lval = mval = val = range;
    }
} *root;

int main() {
    FILE *fin = fopen("roadwork.in", "r"), *fout =
fopen("roadwork.out", "w");

    int N, L, X;
    fscanf(fin, "%d %d %d", &N, &L, &X);

    root = new Node(1, L, false);
    for(int i=1;i<=N;i++) {
        int lo, hi;
        fscanf(fin, "%d %d", &lo, &hi);
        root->paint(lo, hi);
        if(root->val < X) {
            fprintf(fout, "%d\n", i);
            fclose(fin); fclose(fout);
            return 0;
        }
    }

    fprintf(fout, "-1\n");
    fclose(fin); fclose(fout);
    return 0;
}
```

Θέμα 3ο: Μετατροπή αριθμών [35 Μονάδες]

C++

Δημήτρης Λως

```
#include <cstdio>
#include <cstdlib>
#include <algorithm>
using namespace std;
long long M;
int sm(long long k, long long n, long long L){
   long long pr = 1, sum = 0;
   for (long long i = 0; i!=n && sum < L; i++){
      sum += pr;
      pr *= k;
   }
   if (L > sum) return -1;
   if (L < sum) return 1;
   return 0;
}
long long minim;
long long check(long long sum, long long test){
   long long point, st, en, mn;
   int tmp;
   for (long long i = 1; i!=45; i++){
      //point  = M / i;
      st = 2; en = M+1;
      while(st < en-1){
         //printf("%lld %lld\n", st, en);
         mn = (st+en)/2;
         tmp = sm(mn, i, sum);
         if (tmp==0) {
            if (test < mn) {minim = min(mn, minim);}
            break;
         }
         else if (tmp==1) en = mn-1;
         else st = mn+1;
      }
      if (sm(en, i, sum)==0) if (test < en) minim = min(en,
minim);
      if (sm(st, i, sum)==0) if (test < st) minim = min(st,
minim);
   }
}
long long divi[2000100];
int main(){
   int N;
   printf("hi");
   FILE *fi = fopen("numbase.in", "r");
   fscanf(fi, "%d", &N);
   FILE *fo = fopen("numbase.out", "w");
```

```
    while(N--){
        //printf("hi");
        fscanf(fi, "%lld", &M);
        //printf("%lld\n", M);
        if (M==1) {fprintf(fo, "2\n"); continue; }
        // For one
        long long z, mn, k = 1;
        // Find divisors
        int count = 0;
        while (k*k <= M){
            if (M%k==0) {
                divi[count++] = k;
                divi[count++] = (long long) M / (long long) k;
            }
            k++;
        }
        minim = M+1;
        for (int i = 0; i<count; i++){
            //printf("hi %lld %lld\n", divi[i], M/divi[i]);
            check(divi[i], (long long)M/(long long)divi[i]);
        }
        fprintf(fo, "%lld\n", minim);
    }
    return 0;
}
```

Με τη συνεργασία:

Αριστοτελείου Πανεπιστημίου Θεσσαλονίκης, Εθνικού και Καποδιστριακού Πανεπιστημίου Αθηνών, Εθνικού Μετσόβιου Πολυτεχνείου, Πανεπιστημίου Αιγαίου, Πανεπιστημίου Ιωαννίνων, Πανεπιστημίου Πατρών, Πανεπιστημίου Πειραιώς, ΤΕΙ Αθήνας.