Разбор задач муниципального этапа олимпиады по информатике

1. В магазине (9-11 классы)

Тема: вывод формулы, разбор случаев
Сложность: простая

Рассчитываем количество коробок для обоих вариантов, умножаем на стоимость за коробку и сравниваем. Чтобы найти значение частного, округленного вверх, используем формулу (N+A-1)/A. Так как суммарная стоимость в подздаче

Пример реализации:

```
var a,b,c,d,n,s1,s2:int64;
begin
  read(a,b,c,d,n);
  s1:=((n+a-1) div a)*b;
  s2:=((n+c-1) div c)*d;
  if s1<s2 then
  begin
    writeln('FIRST');
    writeln(s1);
  end
  else if s1>s2 then
  begin
    writeln('SECOND');
    writeln(s2);
  end
  else
  begin
    writeln('ANY');
    writeln(s1);
  end;
end.
```

1. Побег робота-кладовщика (7-8 класс)

Тема: исполнители
Сложность: простая

Примеры программ
Для 1 подзадачи: TFRDRTRDRFTRDRFTRFDRFFTRFDRFFTRFFCCDC
Для 1 и 2 подзадачи: F*TRF*RFRDRF*TRF*DRF*TRF*RFRDRF*TRF*DRF*TRF*RFRDRF*TRF*CCDC

2. Алгоритм (7-8 класс)

Тема: реализация программы по схеме алгоритма
Сложность: простая

Пример реализации:

```
var n,k,m:integer;
begin
  read(n);
  k:=0;
  while n>9 do
  begin
    k:=k+1;
    m:=1;
    while n>0 do
    begin
      m:=m*(n mod 10);
      n:=n div 10;
    end;
    n:=m;
  end;
  writeln(k,' ',n);
end.
```

2. Алгоритм (9 класс)

Тема: реализация программы по схеме алгоритма
Сложность: простая

Пример реализации:

```
var n,a,b,c,i:integer;
begin
  read(n,a,b,c);
  i:=1;
  if not((a+1<>b) and (b+1=c)) then
  begin
    repeat
      i:=i+1;
      if a+1<>b then break;
      a:=b;
      b:=c;
      if i<n-1 then read(c);
    until i>=n;
  end;
  writeln(i);
end.
```

2 (4). Первоклассные числа (9-11 класс)

Тема: динамическое программирование
Частичные решения: полный перебор, выполнение действий по заданному алгоритму
Сложность: выше среднего

Полное решение: является ли число первоклассным, зависит только первоначальной суммы квадратов цифр. Всего есть 18*81 вариант для этой суммы. Можно с помощью динамического программирования подсчитать количество чисел с некоторой суммой, затем просуммировать количества чисел для тех сумм, которые приводят к превосходным числам.
Для подсчета в диапазоне от A до B применяем дважды функцию F(X), которая возвращает количество чисел в диапазоне от 1 до X: F(B)-F(A-1). Подсчет количеств ведем табличным способом для количества разрядов от 1 до 18. Для обработки границы рассматриваем начало числа и изменяем следующий за этим разряд числа до 0, а количество вариантов для остальных цифр получаем из таблицы.

Пример реализации:

```
const MAXK=18;
      MAXS=MAXK*81;
var a,b:int64;
  i,k,d,d2:integer;
  is1c:array[0..MAXS] of integer;
  dp:array[0..MAXK,0..MAXS] of int64;
function sumdig(n:int64):integer;
var n1,s1:integer;
begin
  s1:=0;
  while n>0 do
  begin
    n1:=n mod 10;
    s1:=s1+n1*n1;
    n:=n div 10;
  end;
  result:=s1;
end;
function is1class(n:integer):boolean;
begin
  if is1c[n]=0 then
  begin
    is1c[n]:=-1;
    if is1class(sumdig(n)) then
      is1c[n]:=1;
  end;
  result:=is1c[n]=1;
```

```
end;
function kol(a:int64):int64;
var k,i,d,sd,s1,d2:integer;
    sum:array[0..MAXS] of int64;
    rez:int64;
begin
  for i:=0 to MAXS do
    sum[i]:=0;
  sum[sumdig(a)]:=1;
  k:=0;
  while a>0 do
  begin
    s1:=a mod 10-1;
    a:=a div 10;
    sd:=sumdig(a);
    for d:=0 to s1 do
    begin
      d2:=d*d;
      for i:=0 to MAXS-sd-d2 do
        sum[i+sd+d2]:=sum[i+sd+d2]+dp[k][i];
    end;
    inc(k);
  end;
  rez:=0;
  for i:=0 to MAXS do
    if is1class(i) then
      rez:=rez+sum[i];
  result:=rez;
end;
begin
  is1c[0]:=-1;
  is1c[1]:=1;
  for i:=0 to MAXS do
    is1class(i);
  dp[0][0]:=1;
  for k:=1 to MAXK do
  begin
    for d:=0 to 9 do
    begin
      d2:=d*d;
      for i:=0 to MAXS-d2 do
        dp[k][i+d2]:=dp[k][i+d2]+dp[k-1][i];
    end;
  end;
  read(a,b);
  writeln(kol(b)-kol(a-1));
end.
```

Пример частичного решения:

```
var a,b,i,k:integer;
function is1class(n:integer):boolean;
var  k,n1,s1:integer;
begin
  k:=0;
  while (n<>1) and (k<250) do
  begin
    k:=k+1;
    s1:=0;
    while n>0 do
    begin
      n1:=n mod 10;
      s1:=s1+n1*n1;
      n:=n div 10;
    end;
    n:=s1;
  end;
```

```
  result:=k<250;
end;
begin
  read(a,b);
  k:=0;
  for i:=a to b do
    if is1class(i) then inc(k);
  writeln(k);
end.
```

3. Первоклассные числа (7-8 класс)

Тема: техника программирования, выполнение действий по заданному алгоритму.
Сложность: простая

Выполнение описанных действий приведет либо к 1, либо будет цикл по другим числам. За 2 шага от числа останется 3 или менее цифры, из которых можно перейти только к числу из 3 или менее цифр, <= $3*9^2=243$. Из принципа Дирихле через 244 шага мы должны попасть на 1 или повторно на какое-то число. Поэтому если за 250 шагов, мы не получаем 1, то число не является первоклассным.

Примеры программ:

```
var s:string;
  k,n,n1,s1,i:integer;
begin
  readln(s);
  k:=1;
  s1:=0;
  for i:=1 to length(s) do
  begin
    n1:=ord(s[i])-ord('0');
    s1:=s1+n1*n1;
  end;
  n:=s1;
  while (n<>1) and (k<250) do
  begin
    k:=k+1;
    s1:=0;
    while n>0 do
    begin
      n1:=n mod 10;
      s1:=s1+n1*n1;
      n:=n div 10;
    end;
    n:=s1;
  end;
  if k>=250 then
    writeln('NO')
  else
    writeln('YES');
end.
```

3. Робот-кладовщик (9 класс)

Тема: техника программирования, моделирование.
Сложность: ниже средней

Пример программы:

```
var p:string;
    i,d,a,n:integer;
    c:boolean;
    h:array[1..20] of integer;
begin
  c:=false;
  d:=1;
  a:=1;
  read(n);
  for i:=1 to n do
```

```
    read(h[i]);
  readln;
  readln(p);
  i:=1;
  while i<=length(p) do
  begin
    if (i<length(p)) and (p[i]='F') and (p[i+1]='*') then
    begin
      while (a+d>=1) and (a+d<=n) and (h[a+d]=h[a]) do
        a:=a+d;
      inc(i);
    end
    else if p[i]='F' then
    begin
      if (a+d>=1) and (a+d<=n) and (h[a+d]=h[a]) then
        a:=a+d;
    end
    else if p[i]='R' then
      d:=-d
    else if p[i]='C' then
    begin
      if (a+d>=1) and (a+d<=n) and ((h[a+d]+1=h[a]) or (h[a+d]-1=h[a])) then
        a:=a+d;
    end
    else if p[i]='T' then
    begin
      if not c and (a+d>=1) and (a+d<=n) and (h[d+a]>0) and
         ((h[a+d]=h[a]+1) or (h[a+d]=h[a]) or (h[a+d]=h[a]+2)) then
      begin
        dec(h[a+d]);
        c:=true;
      end;
    end
    else if p[i]='D' then
    begin
      if c and (a+d>=1) and (a+d<=n) and
          ((h[a+d]+1=h[a]) or (h[a+d]=h[a]) or (h[a+d]-1=h[a])) then
      begin
        inc(h[a+d]);
        c:=false;
      end;
    end;
    inc(i);
  end;
  writeln(a);
  for i:=1 to n do
    write(' ',h[i]);
  writeln;
end.
```

3. Прогулка в лабиринте (10-11 класс)

Тема: рекурсия
Частичные решения: предрешение для подзадачи 1

Сложность: ниже среднего

Частичное решение: для каждого из лабиринтов вычисляем массивы координат x и y для каждого шага. Выводим значение из M-го элемента массива координат, соответствующего N.

Полное решение: вычисляем номер шага в лабиринте меньшей размерности, находим координаты и преобразуем в соответствии со смещение и поворотом.

Пример программы:

```
var n,m,x,y:integer;
procedure calc(n,m:integer;var x,y:integer);
var n2:integer;
begin
```

```
  if n=1 then
  begin
    x:=1;
    y:=1;
    exit;
  end;
  n2:=n div 2;
  if m<=n2*n2 then { левый нижний }
  begin
    calc(n2,m,y,x);
  end
  else if m<=2*n2*n2 then { левый верхний }
  begin
    calc(n2,m-n2*n2,x,y);
    y:=y+n2;
  end
  else if m<=3*n2*n2 then { правый верхний }
  begin
    calc(n2,m-2*n2*n2,x,y);
    y:=y+n2;
    x:=x+n2;
  end
  else
  begin { правый нижний }
    calc(n2,m-3*n2*n2,y,x);
    y:=n2+1-y;
    x:=n+1-x;
  end;
end;
begin
  read(n,m);
  calc(n,m,x,y);
  writeln(x,' ',y);
end.
```

4. Рамки (7-8 класс)

Тема: сортировка, сканирование
Частичные решения: полный перебор для 1 подзадачи, подсчет количеств вместо сортировки в подзадаче 2, сортировка пузырьком в подзадаче 3.

Сложность: выше среднего

```
Полное решение: сортируем последовательность, рассматриваем пары, если можно для
пары найти подходящую по длине пару, то увеличиваем счетчик, иначе заменяем
предыдущую, слишком короткую пару, на новую.

var n,k,p,i:integer;
  o:array[1..100000] of integer;

procedure qsort(l,r:integer);
var i,j,m,t:integer;
begin
  if l>=r then  exit;
  m:=o[l+random(r-l+1)];
  i:=l;
  j:=r;
  while i<=j do
  begin
    while o[i]<m  do  inc (i);
    while o[j]>m  do  dec (j);
    if i<=j then
    begin
      t:=o[i];o[i]:=o[j];o[j]:=t;
      inc(i); dec(j);
    end;
  end;
```

```
  qsort(l,j);
  qsort(i,r);
end;

begin
  read(n);
  for i:=1 to n do
    read(o[i]);
  qsort(1,n);
  p:=0;
  k:=0;
  i:=1;
  while i<n do
  begin
    if (o[i]=o[i+1]) then
    begin
      if p=0 then p:=o[i]
      else
      begin
        if 2*p>=o[i] then
        begin
          inc(k);
          p:=0;
        end
        else p:=o[i];
      end;
      i:=i+2;
    end
    else
      inc(i);
  end;
  writeln(k);
end.
```

4. Скобки (10-11 класс)

Тема: бинараный поиск, дерево отрезков
Частичное решение: жадный алгоритм.
Сложность: высокая

Жадным способом можно легко получить частичное решение — удаляем те закрывающиеся скобки, которые при просмотре слева направо и справа налево загоняют баланс скобок в минус.

Пример программы:

```
#include <bits/stdc++.h>

using namespace std;

const int NMAX = 500000 + 5;
int N, v[NMAX];

struct Node {
    int l, r;
    int sum, minSuf;
    Node(int _l = 0, int _r = 0, int _sum = 0, int _minSuf = 0):
        l(_l), r(_r), sum(_sum), minSuf(_minSuf) {}

    static Node singleNode(const int l, const int r, const int val) {
        return Node(l, r, val, min(val, 0));
    }
    friend Node operator+(const Node &arg0, const Node &arg1) {
        return Node(arg0.l, arg1.r, arg0.sum + arg1.sum, min(arg0.minSuf +
arg1.sum, arg1.minSuf));
    }
} tree[4 * NMAX];
```

```
void build(int node, int l, int r) {
    if (l == r) {
        tree[node] = Node :: singleNode(l, l, v[l]);
        return ;
    }
    const int mid = (l + r) / 2;
    build(2 * node, l, mid);
    build(2 * node + 1, mid + 1, r);
    tree[node] = tree[2 * node] + tree[2 * node + 1];
}

Node query(int node, int l, int r) {
    if (tree[node].l == l && tree[node].r == r)
        return tree[node];
    const int mid = (tree[node].l + tree[node].r) / 2;
    if (r <= mid)
        return query(2 * node, l, r);
    else if (l > mid)
        return query(2 * node + 1, l, r);
    else {
        return query(2 * node, l, mid) +
               query(2 * node + 1, mid + 1, r);
    }
}

void update(int node, int where, int val) {
    if (tree[node].l == tree[node].r) {
        v[where] += val;
        tree[node] = Node :: singleNode(where, where, v[where]);
        return ;
    }
    const int mid = (tree[node].l + tree[node].r) / 2;
    if (where <= mid)
        update(2 * node, where, val);
    else
        update(2 * node + 1, where, val);
    tree[node] = tree[2 * node] + tree[2 * node + 1];
}

struct Query {
    int r, pos;
    Query(int _r = 0, int _pos = 0):
        r(_r), pos(_pos) {}
};
vector <Query> queries[NMAX];
int answers[NMAX];

int main() {
    ios::sync_with_stdio(0);
    cin.tie(0);
    cin >> N;
    string str;
    cin >> str;
    for (int i = 1; i <= N; ++ i) {
        v[i] = str[i - 1] == '(' ? 1 : -1;
    }
    build(1, 1, N);

    int Q = 0;
    cin >> Q;

    for (int i = 1; i <= Q; ++ i) {
        int l, r;
        cin >> l >> r;
        queries[l].push_back(Query(r, i));
```

```
    }

    vector <int> stk;
    for (int i = N; i; -- i) {
        if (v[i] == -1)
            stk.push_back(i), update(1, i, 1);
        else {
            if (!stk.empty())
                update(1, stk.back(), -1), stk.pop_back();
        }
        for (auto qr: queries[i]) {
            const int l = i, r = qr.r;
            answers[qr.pos] = upper_bound(stk.rbegin(), stk.rend(), r) -
stk.rbegin() - query(1, l, r).minSuf;
        }
    }

    for (int i = 1; i <= Q; ++ i)
        cout << answers[i] << '\n';
    return 0;
}
```