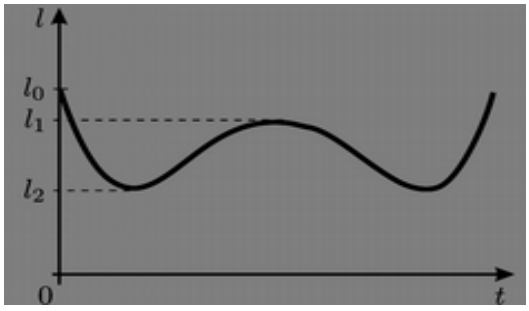
```
{
    vector<int> primes;
    prime(primes);

    int k;
    scanf("%d\n", &k);

    for (int i = 0; i < k; i++)
    {
        int n;
        scanf("%d\n", &n);
        printf("%d\n", primes[n - 1]);
    }

    return 0;
}
```

§2 Второй отборочный этап

2.1 Задачи по физике

Задача 2.1.1 (5 баллов)

Условие:



С помощью лазерного дальномера была записана информация о расстоянии до тела, брошенного вертикально вверх с поверхности земли. Зависимость расстояния *l(t)* между этим телом и неподвижным наблюдателем приведена на графике. На какой высоте над землёй находится наблюдатель? На каком расстоянии от линии, по которой движется тело, находится наблюдатель? Чему равна начальная скорость тела? Величины l_0=10м, l_1=8м и l_2=4м, ускорение свободного падения g=10м/с^2. Ответ округлите до целых.

Решение:

Пусть наблюдатель находится на высоте h и на расстоянии a от линии, по которой движется тело. При бросании тела возможны два случая:

1. тело не долетает до высоты, на которой находится наблюдатель – в этом случае расстояние l от тела до наблюдателя сначала уменьшается, а затем увеличивается;
2. тело поднимается выше наблюдателя – в этом случае расстояние l сначала уменьшается от $\sqrt{a^2+h^2}$ до a, затем увеличивается до $\sqrt{a^2+(H-h)^2}$, где H – высота подъёма тела, а потом опять уменьшается до a и увеличивается до $\sqrt{a^2+h^2}$.

Как видно из приведённого в условии рисунка, реализуется именно второй случай. При этом

$$l_0=\sqrt{a^2+h^2},\ l_1=\sqrt{a^2+(H-h)^2}.$$

Отсюда находим: a=l_2,

$$h=\sqrt{l_0^2-a^2}=\sqrt{l_0^2-l_2^2}=9\,м$$

и

$$H=h+\sqrt{l_1^2-a^2}=\sqrt{l_0^2-l_2^2}+\sqrt{l_1^2-l_2^2}=16\,м.$$

Начальную скорость тела можно определить из соотношения

$$v_0^2 = 2gH \rightarrow v_0 = \sqrt{2gH} = 18\frac{м}{с}.$$

Ответ:

1. наблюдатель находится на высоте h=9м над землёй;
2. наблюдатель находится на расстоянии $a=l_2$=4м от линии, по которой движется тело;
3. начальная скорость тела равна v_0=18м/с.

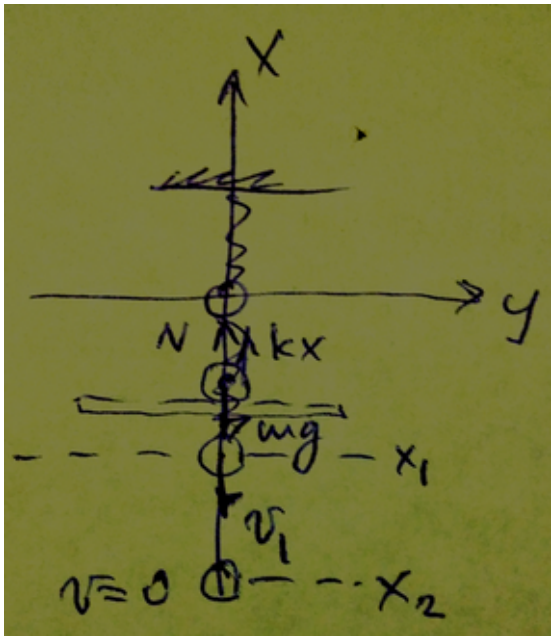
Задача 2.1.2 (4-7 баллов)

Условие:

Груз массы m=1кг подвешенный на идеальной пружине жёсткостью k=100Н/м удерживается с помощью горизонтальной подставки так, что пружина не деформирована. В некоторый момент времени подставку начинают двигать вертикально вниз с ускорением a. Найти максимальное смещение груза из начального положения. Рассмотреть два случая: a_1=12м/с^2 и a_2=2м/с^2. Ускорение свободного падения принять равным 10 м/с^2, ответ дать в сантиметрах.

Решение:

Выберем СО так, что ось координат направлена вверх и ноль совпадает с начальным положением груза. Рассмотрим случай, когда ускорение подставки $a<g$. Запишем второй закон Ньютона в проекции на вертикальную ось:



$$mg - kx - N = ma.$$

Условие отрыва тела от опоры: N=0. Следовательно:

$$k x_1 = m(g - a),$$

где x_1 – координата тела в момент отрыва от подставки. К этому моменту времени тело наберёт скорость v_1:

$$v_1 = \sqrt{2 a x_1}$$

и продолжит движение вниз до полной остановки за счёт силы реакции пружинки. Пусть в точке x_1 потенциальная энергия тела в поле сил тяжести Земли равна нулю. По закону сохранения полной механической энергии:

$$\frac{m v_1^2}{2} + \frac{k x_1^2}{2} = \frac{k x_2^2}{2} - mg(x_2 - x_1).$$

Найдём координату тела в момент остановки:

$$x_2 = \frac{mg \pm \sqrt{(mg - k x_1)^2 + km v_1^2}}{k}.$$

Подставим выше полученные выражения для x_1 и v_1 и упростим, получим:

$$x_2 = \frac{m}{k}\left(g \pm \sqrt{a(2g - a)}\right). \qquad (*)$$

Движение тела будут колебательным относительно положения равновесия mg/k с амплитудой $\frac{\sqrt{a(2g-a)}}{k}$. Соответственно низшему положению тела во время движения будет соответствовать знак плюс в выражении (*).

В случае $a>g$ груз оторвётся от поверхности сразу после начала движения подставки. Тогда из закона сохранения энергии:

$$0=-mgx+\frac{kx^2}{2}$$

найдём низшую точку:

$$x=2g\frac{m}{k}.$$

Ответ:

$$a_1\geq g, x_1=2m\frac{g}{k}=20\,см; a_2<g, x_2=m\frac{\left(g+\sqrt{a_2(2g-a_2)}\right)}{k}=16\,см.$$

Задача 2.1.3 (5 баллов)

Условие:

В закрытом сосуде находится 3 литра воды. Воду довели до температуры 120^0C. После этого прекратили нагрев и через небольшие отверстия спустили пар. Сколько воды останется в сосуде после того, как вода перестанет кипеть? Удельные теплота парообразования и теплоёмкость воды соответственно равны L=2,2МДж/кг и C=4,2кДж/(кг^{0}C). Теплоёмкостью стенок сосуда и потерями тепла через них пренебречь. Ответ дайте в килограммах и округлите до десятых.

Решение:

После открывания клапана давление в скороварке упадёт, и вода будет кипеть, остывая при этом от $T_1 = 120^0C$ до $T_2 = 100^0C$. Количество теплоты, необходимое для испарения, будет отниматься от остывающей воды. Пусть за малый промежуток времени испарилась небольшая масса воды Δm_i, а температура воды при этом уменьшилась на ΔT_i. Тогда уравнение теплового баланса имеет вид:

$$Cm_i\Delta T_i=L\Delta m_i,$$

где m_i – масса воды, которая была в скороварке при испарении массы Δm_i. Отсюда

$$\Delta T_i=\frac{L}{C}\frac{\Delta m_i}{m_i},$$

и

$$\Delta T=T_1-T_2=\frac{L}{C}\sum_i\frac{\Delta m_i}{m_i}.$$

Сумму в последней формуле можно точно вычислить при помощи интегрирования, а можно ограничиться лишь её оценкой. Пусть M_0 = 3 кг – начальная масса воды в скороварке, $M_{ост}$ – масса воды, оставшейся в скороварке после окончания процесса кипения. Заметим, что

$$\sum_i\Delta m_i=M_0-M_{ост}$$

– масса выкипевшей воды. Тогда

$$T_1-T_2=\frac{L}{C}\sum_i\frac{\Delta m_i}{m_i}>\frac{L}{CM_0}\sum_i\Delta m_i=\frac{L}{C}\frac{M_0-M_{ост}}{M_0},$$

откуда

$$M_{ост}>M_0\left(1-\frac{C(T_1-T_2)}{L}\right)\approx 2{,}886\,кг.$$

Это нижняя оценка для массы оставшейся в скороварке воды.

Аналогично можно получить для $M_{ост}$ оценку сверху:

$$T_1 - T_2 = \frac{L}{C}\frac{\sum_i \square\,\Delta m_i}{m_i} < \frac{L}{C}\frac{\sum_i \square\,\Delta m_i}{M_0 - \sum_i \square\,\Delta m_i} = \frac{L}{C}\frac{M_0 - M_{ост}}{M_{ост}},$$

откуда

$$M_{ост} < M_0 \frac{L}{L + C(T_1 - T_2)} \approx 2{,}890\,кг.$$

Видно, что верхняя и нижняя оценки очень близки. Поэтому можно принять в качестве ответа среднее арифметическое из них: Мост approx 2,888кг. Следует отметить, что точный ответ Мост approx 2,8876кг, полученный при помощи интегрирования, отличается от найденной нами величины менее чем на 0,02 процента.

Ответ: 2,8 кг.

Задача 2.1.4 (5 баллов)

Условие:

В качестве элемента сенсора температуры используется конденсатор, заполненный веществом с удельным сопротивлением ρ зависящим от напряжённости E электрического поля между обкладками по следующему закону: $\rho = 3\cdot 10^{-7} + 2\cdot 10^{-3} E^2$. Площадь пластин S=1см^2, расстояние меду ними d=1мм. Найдите максимальное значение тока I_{max} через конденсатор. Определите зависимость мощности тока от напряжения между пластинами и найдите максимально возможное $P(U)$. Ответ приведите в амперах и микроваттах.

Решение:

Найдём максимальную силу тока двумя способами.

Первый способ

$$\rho = \rho_0 + A E^2 = A\left(\frac{\rho_0}{A} + E^2\right)$$

Сила тока:

$$I = \frac{U}{R} = \frac{US}{\rho d} = \frac{EdS}{A\left(\frac{\rho_0}{A} + E^2\right)d}.$$

Выразим отсюда E:

$$E^2 - E\frac{S}{IA} + \frac{\rho_0}{A} = 0.$$

Отсюда:

$$E_{1,2} = \frac{S}{2IA} \pm \sqrt{\left(\frac{S}{2IA}\right)^2 - \frac{\rho_0}{A}}.$$

Последнее выражение имеет вещественные корни, если дискриминант положителен, тогда

$$\left(\frac{S}{2IA}\right)^2 - \frac{\rho_0}{A} \geq 0$$

и

$$I_{max} = \frac{S}{2\sqrt{\rho_0 A}} = 2\,A.$$

Второй способ

По закону Ома:

$$I=\frac{U}{R}=\frac{US}{\rho d}=\frac{EdS}{\left(\rho_0+A\,E^2\right)d}.$$

Найдём производную по E:

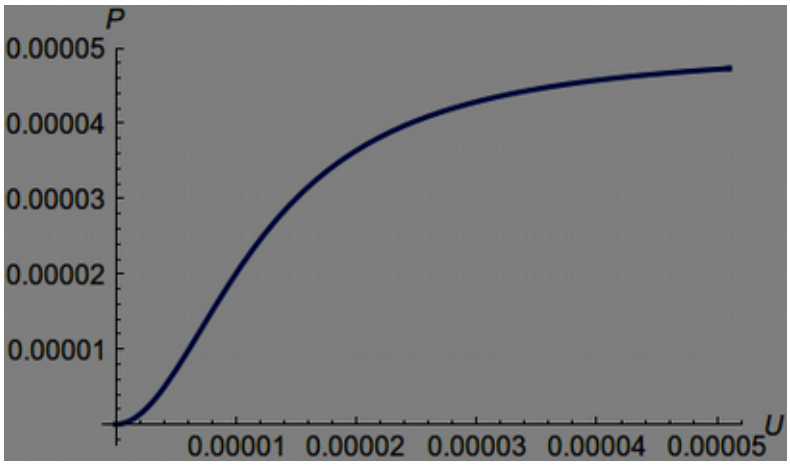
$$\frac{dI}{dE}=S\frac{\rho_0+A\,E^2-E\,2\,AE}{\left(\rho_0+A\,E^2\right)^2}=0, E=\left(\frac{\rho_0}{A}\right)^{\frac{1}{2}}.$$

Подставим в выражение для тока

$$I_{max}=\frac{S\left(\frac{\rho_0}{A}\right)^{\frac{1}{2}}}{\rho_0+A\frac{\rho_0}{A}}=\frac{S}{2\sqrt{\rho_0 A}}=2\,A.$$

Найдём теперь мощность:

$$P=\frac{U^2}{R}=\frac{U^2 S}{\rho d}=\frac{U^2 S}{\left(\rho_0+A\,E^2\right)d}=\frac{S}{d}\frac{U^2}{\rho_0+A\frac{U^2}{d^2}}.$$



При малых напряжениях график ведёт себя как парабола, при $U\to\infty$:

$$\lim_{\square}\square_{U\to\infty}\frac{S}{d}\frac{1}{\frac{\rho_0}{U^2}+\frac{A}{d^2}}=\frac{Sd}{A}=50\,мкВт.$$

$$\lim_{\square}\square_{U\to\infty}P=¿\lim_{\square}\square$$

Ответ:

I_{max} = 2 А, P = 50 мкВт.

2.2 Задачи по информатике

Задача 2.2.1 (10 баллов)

Условие:

Одним из элементов умного дома является система контроля климата. Эта система нацелена на поддержание оптимального значения температуры в комнате. Температурная шкала в рамках данной системы разбивается на 5 возможных состояний:

- colder – очень холодно
- cold – холодно
- warm – тепло
- hot – жарко

- hotter – очень жарко

В зависимости от температуры, полученной от термодатчика, система переводится в одно из этих состояний. Для заданной температуры t и набора заданных граничных значений t_{colder}, t_{cold}, t_{hot}, t_{hotter} состояние системы определяется по следующим правилам:

- colder – если $t < t_{colder}$
- cold – если $t_{colder} \le t < t_{cold}$
- warm – если $t_{cold} \le t \le t_{hot}$
- hot – если $t_{hot} < t \le t_{hotter}$
- hotter – если $t_{hotter} < t$

Ваша задача определить в каком из состояний система находилась в каждый момент времени.

Формат входных данных

В первой строке даны 4 числа t_{colder}, t_{cold}, t_{hot}, t_{hotter} – граничные значения температур для определения состояния ($-40.0 \le t_{colder} < t_{cold} < t_{hot} < t_{hotter} \le 40.0$).
Во второй строке дано целое число N – количество значений температуры, для которых необходимо определить состояние системы ($1 \le N \le 1000$).
В следующих N строках содержится по одному числу t_i – значение температуры, полученное от термодатчика ($-50.0 \le t_i \le 50.0$).
Все температурные значения гарантировано содержат один знак после десятичной точки.

Формат выходных данных

Выведите N строк, каждая из которых содержит одно из слов “colder”, “cold”, “warm”, “hot”, “hotter” (без кавычек), соответствующих состоянию системы для всех t_i (порядок вывода должен соответствовать входному порядку t_i).

Пример

Входные данные	**Выходные данные**
8.5 15.0 27.5 32.1 5 7.0 9.0 16.5 28.0 33.0	colder cold warm hot hotter

Решение:

Данная задача проверяет умение участников использовать условные операторы в используемых ими языках программирования. Для решения данной задачи необходимо аккуратно реализовать проверку всех условий перехода системы в каждое из описанных состояний. Для простоты сравнения температурных значений, являющихся вещественными числами с фиксированной точкой, рекомендуется приводить их к целому типу, путём сдвига запятой на 1 знак вправо.

Код решения (C++)

```
#define _CRT_SECURE_NO_WARNINGS
```

```
#pragma comment(linker, "/STACK:4000000")

#include <iostream>
#include <cstdio>
#include <string>
#include <vector>
#include <utility>
#include <algorithm>
#include <ctime>

using namespace std;

int to_fixed(int t1, int t2)
{
    return t1 < 0 ? t1 * 10 - t2 : t1 * 10 + t2;
}

string int2str(int x)
{
    string res = "";
    bool neg = false;
    if (x < 0)
    {
        neg = true;
        x *= -1;
    }
    do
    {
        res = char((x % 10) + '0') + res;
    } while (x /= 10);

    if (neg)
    {
        res = "-" + res;
    }

    return res;
}

string to_fixed_string(int x)
{
    string res = int2str(x / 10) + "." + int2str(abs(x % 10));

    return res;
}

int main()
{
    int tcc, tc, th, thh;
    int t1, t2, t3, t4;
```

```
      scanf("%d.%d %d.%d %d.%d %d.%d\n", &tcc, &t1, &tc, &t2,
&th, &t3, &thh, &t4);
    tcc = to_fixed(tcc, t1);
    tc = to_fixed(tc, t2);
    th = to_fixed(th, t3);
    thh = to_fixed(thh, t4);

    int n;
    scanf("%d\n", &n);

    for (int i = 0; i < n; i++)
    {
        scanf("%d.%d\n", &t1, &t2);
        t1 = to_fixed(t1, t2);
        if (t1 < tcc)
        {
            printf("colder\n");
        }
        else if (tcc <= t1 && t1 < tc)
        {
            printf("cold\n");
        }
        else if (tc <= t1 && t1 <= th)
        {
            printf("warm\n");
        }
        else if (th < t1 && t1 <= thh)
        {
            printf("hot\n");
        }
        else if (thh < t1)
        {
            printf("hotter\n");
        }
    }

}
```

Задача 2.2.2 (15 баллов)

Условие:

Система контроля климата умного дома имеет возможность включать и выключать систему отопления или охлаждения в зависимости от температурного состояния, описанного в прошлой задаче. Алгоритм принятия решения о включении или выключении той или иной системы следующий:

- Если включена система охлаждения и система управления переходит из одного из состояний warm, hot или hotter в состояние cold, при этом состояние cold сохраняется хотя бы три временных отсчёта, то система охлаждения отключается.

- Если система переходит из одного из состояний cold, warm, hot или hotter в состояние colder, при этом состояние colder сохраняется хотя бы три временных отсчёта, то отключается система охлаждения (в случае, если она была включена), после чего включается система отопления (в случае, если она была выключена).
- Если включена система отопления и система управления переходит из одного из состояний warm, cold или colder в состояние hot, при этом состояние hot сохраняется хотя бы три временных отсчёта, то система отопления отключается.
- Если система переходит из одного из состояний hot, warm, cold или colder в состояние hotter, при этом состояние hotter сохраняется хотя бы три временных отсчёта, то отключается система отопления (в случае, если она была включена), после чего включается система охлаждения (в случае, если она была выключена).

Ваша задача по заданному списку состояний системы в каждый временной момент определить, когда необходимо включить/выключить каждую из систем – отопления или охлаждения. В начальный момент времени все устройства отключены и система находится в состоянии warm.

Формат входных данных
В первой строке дано целое число N – количество временных отсчётов ($1 \le N \le 1000$).
В следующих N строках содержится одно из слов “colder”, “cold”, “warm”, “hot”, “hotter” (без кавычек), описывающих состояние системы в каждый момент времени.

Формат выходных данных
Выведите N строк, каждая из которых содержит описание действий, которые необходимо выполнить в каждый момент времени.
Действия описываются командами вида: “<system_name> <action>” (без кавычек), где <system_name> – название системы (heater – отопление, cooler – охлаждение); <action> – действие включения или выключения (on – включить, off – выключить). Если в один момент времени необходимо выполнить более одного действия, необходимо разделить их символом “;” и последующим пробелом. Если никаких действий выполнять не требуется, выведите слово “none” без кавычек.

Пример

Входные данные	**Выходные данные**
8 warm hotter hotter hotter hotter colder colder colder	none none none cooler on none none none cooler off; heater on

Решение:

При решении данной задачи необходимо рассмотреть все возможные варианты развития событий и правильно написать сценарий переключения состояний охладителя и нагревателя. Стоит обратить внимание, что возможны случаи, когда система сохраняет состояние cold в течение трёх и более отсчетов времени при включенном обогревателе, но обогреватель не будет выключен. Аналогично при включенном охладителе и не менее трёх отсчётах состояния hot охладитель может не быть выключен никогда.

Код решения (C++)

```
#define _CRT_SECURE_NO_WARNINGS
#pragma comment(linker, "/STACK:4000000")

#include <iostream>
#include <cstdio>
#include <string>
#include <vector>
#include <utility>
#include <algorithm>
#include <ctime>

using namespace std;

int main()
{
    int n;
    scanf("%d\n", &n);
    char s[100] = { 0 };
    bool cooler = false;
    bool heater = false;
    int cc = 0;
    int c = 0;
    int h = 0;
    int hh = 0;
    bool from_hot = false;
    bool from_cold = false;

    for (int i = 0; i < n; i++)
    {
        scanf("%s\n", s);
        string str(s);
        string res = "";
        if (str == "colder")
        {
            ++cc;

            if (cc == 3) {
                if (cooler)
                {
                    cooler = false;
                    res += "cooler off";
                }
```

```
        if (!heater)
        {
            heater = true;
            if (res.length())
            {
                res += "; ";
            }
            res += "heater on";
        }
    }

    c = 0;
    h = 0;
    hh = 0;
}
else if (str == "hotter")
{
    ++hh;

    if (hh == 3) {
        if (heater)
        {
            heater = false;
            res += "heater off";
        }
        if (!cooler)
        {
            cooler = true;
            if (res.length())
            {
                res += "; ";
            }
            res += "cooler on";
        }
    }

    cc = 0;
    c = 0;
    h = 0;
}
else if (str == "warm")
{
    cc = 0;
    c = 0;
    h = 0;
    hh = 0;
}
else if (str == "cold")
{
    c++;
```

```
    if (c == 1)
    {
        if (h || hh || !cc)
        {
            from_hot = true;
        }
        else
        {
            from_hot = false;
        }
    }

    if (c == 3 && from_hot)
    {
        if (cooler)
        {
            cooler = false;
            res += "cooler off";
        }
    }

    cc = 0;
    h = 0;
    hh = 0;
}
else if (str == "hot")
{
    h++;
    if (h == 1)
    {
        if (c || cc || !hh)
        {
            from_cold = true;
        }
        else
        {
            from_cold = false;
        }
    }

    if (h == 3 && from_cold)
    {
        if (heater)
        {
            heater = false;
            res += "heater off";
        }
    }

    hh = 0;
```

```
            c = 0;
            cc = 0;
        }

        if (!res.length())
        {
            res = "none";
        }

        printf("%s\n", res.c_str());
    }
}
```

Задача 2.2.3 (15 баллов)

Условие:

Система контроля климатом умного дома опрашивает температурный датчик с частотой 1 раз в секунду. На основании полученного значения температуры система переходит в одно из состояний, описанных в задаче 1. От динамики изменения состояний работа отопительных или охладительных приборов определяется в задаче 2. В рамках этой задачи вам предлагается соптимизировать процесс принятия решения о включении и выключении этих приборов.

Для простоты будем полагать, что при работающем отопительном или охладительном приборе температура в комнате меняется по линейному закону. В один момент времени может работать либо отопительный, либо охладительный прибор, либо все приборы отключены. Другими словами, в каждый момент времени температура в комнате либо увеличивается со скоростью T_{heater} градусов в секунду (включен отопительный прибор), либо уменьшается со скоростью T_{cooler} градусов в секунду (включен охладительный прибор), либо не меняется вовсе (все приборы выключены).

Ваша задача по текущей температуре T_0 и заданным параметрам T_{heater} и T_{cooler} определить какой прибор, отопительный или охладительный, необходимо включить и на какое время, чтобы температура в комнате стала равна заданной T_t .

Формат входных данных

В единственной строке даны четыре вещественных положительных числа T_0 , T_{heater} , T_{cooler} и T_t . Все числа по абсолютной величине не превышают значение 50. Гарантируется, что T_0 отличается от T_t не менее чем на одну десятую; величины T_{heater} и T_{cooler} всегда положительные.

Формат выходных данных

В единственной строке через пробел выведите тип прибора (“heater” – отопительный, “cooler” – охладительный, без кавычек), который необходимо включить, и время в секундах, с точностью не менее двух знаков после запятой, которое этот прибор должен проработать, чтобы температура в комнате стала равна T_t .

Пример

Входные данные	**Выходные данные**
13.5 0.125 0.25 11.8125	cooler 6.75

Решение:

Для определения типа прибора, который необходимо включить, необходимо сравнить величины T_0 и T_t. Если $T_0<T_t$, необходимо включить отопительный прибор, иначе – охладительный. Время t_h, на которое необходимо включить отопительный прибор расчитывается по формуле:

$$t_h=\frac{T_t-T_0}{T_{heater}}.$$

Время t_c, на которое необходимо включить охладительный прибор расчитывается по формуле:

$$t_c=\frac{T_0-T_t}{T_{cooler}}.$$

Код решения (C++)

```
#define _CRT_SECURE_NO_WARNINGS
#pragma comment(linker, "/STACK:4000000")

#include <iostream>
#include <cstdio>
#include <string>
#include <vector>
#include <utility>
#include <algorithm>
#include <ctime>

using namespace std;

int main()
{
    double t0, th, tc, tt;

    scanf("%lf%lf%lf%lf", &t0, &th, &tc, &tt);

    if (t0 < tt)
    {
        printf("heater %.2lf\n", (tt - t0) / th);
    }
    else
    {
        printf("cooler %.2lf\n", (t0 - tt) / tc);
    }
}
```

Задача 2.2.4 (20 баллов)

Условие:

Для более точного определения момента включения или выключения отопительных или охладительных приборов системы контроля климата умного дома было решено

использовать K температурных датчика вместо одного. По аналогии с задачей 1 каждому термодатчику ставится в соответствие одно из состояний colder, cold, warm, hot и hotter на основании температуры t_i -го датчика и граничных значений температур t_{colder}, t_{cold}, t_{hot}, t_{hotter}, одинаковых для всех датчиков. По каждому датчику температуры отдельно принимается решение о необходимости включения или выключения отопительной или охладительной системы в зависимости от состояний по правилам, описанным в задаче 2.

Отопительный или охладительный прибор включаются только в том случае, если как минимум C термодатчиков приняли решение о необходимости включения этого устройства.

Ваша задача определить в какие моменты времени и какие виды устройств необходимо включить или выключить. В начальный момент времени все устройства отключены и все термодатчики находятся в состоянии warm.

Формат входных данных

В первой строке даны 4 числа t_{colder}, t_{cold}, t_{hot}, t_{hotter} – граничные значения температур для определения состояния ($-40.0 \le t_{colder} < t_{cold} < t_{hot} < t_{hotter} \le 40.0$).
Во второй строке даны три целых числа N – количество моментов времени, для которых необходимо определить состояние системы ($1 \le N \le 1000$); K – количество термодатчиков ($1 \le K \le 1000$); C – минимальное количество датчиков для принятия решения о включении или выключении того или иного устройства ($\frac{N}{2} < C \le N$).

В следующих N строках содержится K числел $t_{i,j}$ – значения температуры, полученные от термодатчиков в -й момент времени ($-50.0 \le t_i \le 50.0$).

Все температурные значения даются с точностью до десятых градуса.

Формат выходных данных

Выведите N строк, каждая из которых содержит описание действий, которые необходимо выполнить в каждый момент времени.
Действия описываются командами вида: “<system_name> <action>” (без кавычек), где <system_name> – название системы (heater – отопление, cooler – охлаждение); <action> – действие включения или выключения (on – включить, off – выключить). Если в один момент времени необходимо выполнить более одного действия, необходимо разделить их символом “;” и последующим пробелом, при этом действие отключения выполняется первым. Если никаких действий выполнять не требуется, выведите слово “none” без кавычек.

Пример

Входные данные	**Выходные данные**
8.5 15.0 27.5 32.1	none
8 5 3	none
26.3 26.5 26.6 26.5 26.2	none
32.2 32.3 30.3 31.8 31.5	none
32.3 32.2 30.3 32.3 32.9	cooler on
32.2 32.2 31.1 32.2 32.0	none
32.2 31.9 31.8 32.4 31.8	none
7.6 7.2 7.5 7.3 7.5	cooler off; heater on
7.7 7.8 7.2 7.5 7.7	
7.8 7.6 7.3 7.5 7.9	

Решение:

Для решения данной задачи необходимо выполнить слияние решений трёх предыдущих задач, при этом необходимо дополнительно правильно считать количество датчиков, принявших решений о включении/выключении отопительного или охладительного прибора. Важно учитывать случаи, когда датчики могут не принять решение об отключении охладителя или обогревателя при сохранении состояний cold и hot соответственно в течение трёх и более отсчётов времени.

Код решения (C++)

```
#define _CRT_SECURE_NO_WARNINGS
#pragma comment(linker, "/STACK:4000000")

#include <iostream>
#include <cstdio>
#include <string>
#include <vector>
#include <utility>
#include <algorithm>
#include <ctime>

using namespace std;

int to_fixed(int t1, int t2)
{
    return t1 < 0 ? t1 * 10 - t2 : t1 * 10 + t2;
}


typedef enum {
    WARM = 0,
    COLD,
    COLDER,
    HOT,
    HOTTER
} State;

State get_state(int t1, int tcc, int tc, int th, int thh)
{
    if (t1 < tcc)
    {
        return COLDER;
    }
    else if (tcc <= t1 && t1 < tc)
    {
        return COLD;
    }
    else if (tc <= t1 && t1 <= th)
    {
```

```
        return WARM;
    }
    else if (th < t1 && t1 <= thh)
    {
        return HOT;
    }
    else if (thh < t1)
    {
        return HOTTER;
    }
}


typedef struct {
    bool cooler = false;
    bool heater = false;
    int cc = 0;
    int c = 0;
    int h = 0;
    int hh = 0;
    bool from_hot = false;
    bool from_cold = false;

    void add_val(State st)
    {
        if (st == COLDER)
        {
            ++cc;

            if (cc == 3) {
                if (cooler)
                {
                    cooler = false;
                }
                if (!heater)
                {
                    heater = true;
                }
            }

            c = 0;
            h = 0;
            hh = 0;
        }
        else if (st == HOTTER)
        {
            ++hh;

            if (hh == 3) {
                if (heater)
```

```
        {
            heater = false;
        }
        if (!cooler)
        {
            cooler = true;
        }
    }

    cc = 0;
    c = 0;
    h = 0;
}
else if (st == WARM)
{
    cc = 0;
    c = 0;
    h = 0;
    hh = 0;
}
else if (st == COLD)
{
    c++;
    if (c == 1)
    {
        if (h || hh || !cc)
        {
            from_hot = true;
        }
        else
        {
            from_hot = false;
        }
    }

    if (c == 3 && from_hot)
    {
        if (cooler)
        {
            cooler = false;
        }
    }

    cc = 0;
    h = 0;
    hh = 0;
}
else if (st == HOT)
{
    h++;
```

```
            if (h == 1)
            {
                if (c || cc || !hh)
                {
                    from_cold = true;
                }
                else
                {
                    from_cold = false;
                }
            }

            if (h == 3 && from_cold)
            {
                if (heater)
                {
                    heater = false;
                }
            }

            hh = 0;
            c = 0;
            cc = 0;
        }
    }
} Therm;

#define GET_STATE(t) get_state(t, tcc, tc, th, thh)

int main()
{

    int tcc, tc, th, thh;
    int t1, t2, t3, t4;

     scanf("%d.%d %d.%d %d.%d %d.%d\n", &tcc, &t1, &tc, &t2,
&th, &t3, &thh, &t4);
    tcc = to_fixed(tcc, t1);
    tc = to_fixed(tc, t2);
    th = to_fixed(th, t3);
    thh = to_fixed(thh, t4);

    int n, k, c;
    scanf("%d%d%d\n", &n, &k, &c);

    vector <Therm> ar(k);

    bool cooler = 0;
    bool heater = 0;
```

```
    for (int i = 0; i < n; i++)
    {
        int cnt_heater = 0;
        int cnt_cooler = 0;
        for (int j = 0; j < k; j++)
        {
            scanf("%d.%d", &t1, &t2);
            t1 = to_fixed(t1, t2);
            State st = GET_STATE(t1);
            ar[j].add_val(st);

            cnt_heater += ar[j].heater;
            cnt_cooler += ar[j].cooler;
        }
        scanf("\n");

        string str = "";

        if (cooler && (k - cnt_cooler) >= c)
        {
            cooler = false;
            str = "cooler off";
        }
        if (heater && (k - cnt_heater) >= c)
        {
            heater = false;
            str = "heater off";
        }
        if (!cooler && cnt_cooler >= c)
        {
            cooler = true;
            if (str.length())
                str += "; ";
            str += "cooler on";
        }
        if (!heater && cnt_heater >= c)
        {
            heater = true;
            if (str.length())
                str += "; ";
            str += "heater on";
        }

        if (!str.length())
            str = "none";

        printf("%s\n", str.c_str());
    }

}
```

Задача 2.2.5 (12 баллов)

Условие:

Как и в любой сложной системе, в системе управления умного дома могут возникать ошибки. Например, в задаче 2 возможны случаи, когда при холодных состояниях системы охладитель не перестанет работать, что является некорректным поведением. Одним из инструментов для поиска ошибок является логирование. Логирование подразумевает сбор информации о происходящих во время работы системы событиях. Система умного дома устроена таким образом, что от устройств на сервер могут поступать различные значения (например, значение температуры от термодатчика) или сервер может отправлять различные значения устройствам (например, необходимый уровень освещения для светильника). Все эти команды при приёме и передаче с точки зрения сервера выглядят следующим образом “<ID> <OP_ID> <VAL>”, где <ID> – числовой идентификатор устройства; <OP_ID> – числовой идентификатор выполняемой операции; <VAL> – числовое значение, отправляемое или принимаемое сервером.

Каждое устройство имеет числовой идентификатор ID и название в виде текстовой строки NAME, состоящей из строчных букв латинского алфавита. Каждое устройство может выполнять различные операции, которые так же имеют числовой идентификатор OP_ID и текстовое описание OP_NAME, состоящее из строчных букв латинского алфавита, описывающее эту операцию. Все значения, которыми оперируют устройства, являются целыми числами.

Ваша задача по полученным сообщениям вида “<ID> <OP_ID> <VAL>” сформировать словесное описание данной команды, которое запишется в лог событий.

Формат входных данных

В первой строке даны два числа: K – количество различных устройств ($1 \le K \le 1000$); O – количество различных операций, которые могут выполнять устройства ($1 \le O \le 1000$); C – количество команд для логирования ($1 \le C \le 1000$).
В следующих K строках содержится целое число ID_i и строка $NAME_i$ – числовой идентификатор -го устройства и его текстовое название соответственно ($1 \le ID_i \le K$). Длина имени $NAME_i$ не превышает 1024 символа.

В следующих O строках содержится целое число OP_{IDi} и строка OP_{NAMEi} – числовой идентификатор -ой операции и её текстовое описание соответственно ($1 \le OP_{IDi} \le O$). Длина OP_{NAMEi} не превышает 7 символов.

В следующих C строках даны три целых числа, описывающих команду: ID_i – идентификатор устройства, задействованного в -ой операции; OP_{IDi} – идентификатор операции, выполняемой i -ой командой; VAL_i – целочисленное значение, операнд данной команды ($-500 \le VAL_i < 500$).

Формат выходных данных

Выведите C строк, каждая из которых содержит описание действий, выполняемых каждой из команд. Формат описания следующий: “<OP_NAME$_i$> <NAME$_i$> <VAL>”.

Пример

Входные данные	**Выходные данные**
3 2 5 2 thermometer 1 heater 3 cooler	receive thermometer 25 receive thermometer 28 send heater 0 receive thermometer 32

1 send 2 receive 2 2 25 2 2 28 1 1 0 2 2 32 3 1 1	send cooler 1

Решение:

Для быстрого поиска названия операции и названия устройства предлагается хранить их в массивах строк, где индексом является идентификатор операции или устройства. Для каждой считанной строки с поступающей командой остаётся выбрать названия операции и устройства из соответствующего массива и вывести требуемую строку.

Код решения (C++)

```
#define _CRT_SECURE_NO_WARNINGS
#pragma comment(linker, "/STACK:4000000")

#include <iostream>
#include <cstdio>
#include <string>
#include <vector>
#include <utility>
#include <algorithm>
#include <ctime>

using namespace std;

typedef struct {
    int id;
    string name;
} IdName;

typedef struct {
    int dev_id;
    int op_id;
    int val;
} Cmd;

int main()
{
    int k, o, c;
    scanf("%d%d%d\n", &k, &o, &c);

    vector <IdName> dev(k + 1);
    vector <IdName> ops(o + 1);
```

```
    for (int i = 0; i < k; ++i)
    {
        char str[1024];
        int id;
        scanf("%d %s\n", &id, str);
        dev[id] = IdName{ id, str };
    }
    for (int i = 0; i < o; ++i)
    {
        char str[1024];
        int id;
        scanf("%d %s\n", &id, str);
        ops[id] = IdName{ id, str };
    }
    for (int i = 0; i < c; ++i)
    {
        int id, oid, val;
        scanf("%d%d%d\n", &id, &oid, &val);
                printf("%s  %s  %d\n",  ops[oid].name.c_str(),
dev[id].name.c_str(), val);
    }
}
```

Задача 2.2.6 (15 баллов)

Условие:

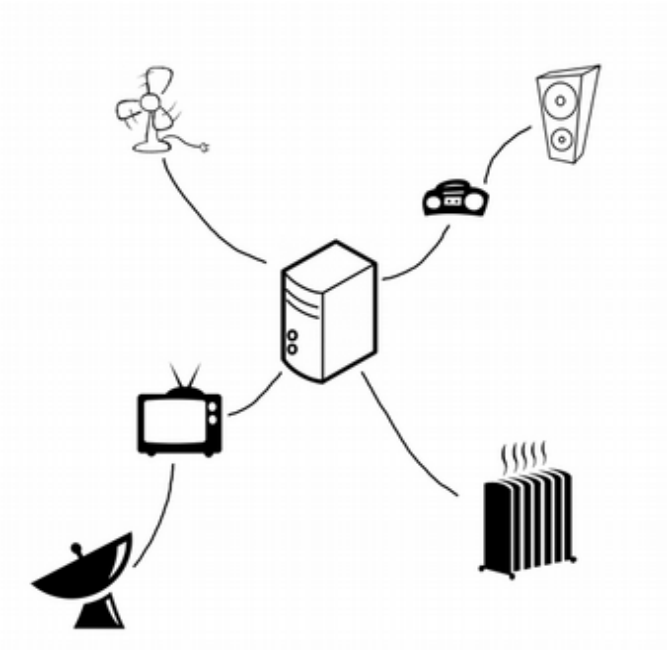
Ограничение по времени: 1 секунда

Ограничение по памяти: 32 мегабайта

Как уже неоднократно упоминалось, все устройства в умном доме соединены в единую сеть. В одной из задач первого отборочного этапа вам уже доводилось обеспечить сеть наименьшим возможным числом связей между устройствами таким образом, чтобы каждый прибор мог связаться с любым другим прибором напрямую или через некоторое количество промежуточных устройств.

Определим порядок подключения N различных устройств в подобную сеть. Пусть каждое устройство имеет свой порядковый номер id, представленный целым числом от 1 до N. id равный 1 всегда принадлежит центральному серверу, который и управляет всей сетью. Изначально все устройства отсоединены друг от друга. Таким образом, сеть состоит только из одного узла – сервера с $id=1$. Устройства подключаются к сети в порядке возрастания номера id. Очередное подключаемое устройство может быть подключено ровно к одному из устройств, уже входящих в сеть. Например, устройство с $id=2$ может быть подключено только к серверу с $id=1$ (т.к. изначально сеть состоит только из сервера); устройство с $id=3$ уже может быть подключено как к серверу с $id=1$, так и к устройству с $id=2$, и т.д.

С целью оптимизации нагрузки на сеть возникла необходимость переместить сервер в самый *центр* полученной сети. Центром сети будем называть такое устройство, которое имеет наименьшее число промежуточных связей до самого удалённого от него устройства.



На рисунке центром сети является сам сервер, т.к. он имеет две промежуточные связи до самого отдалённого от него устройства (колонки или спутниковой тарелки), а все другие устройства имеют как минимум три промежуточных связи до самых отдаленных устройств (например, от телевизора дальше всего расположены колонки – 3 промежуточные связи).
Ваша задача определить номера устройств, являющихся центрами сети по заданному порядку подключения устройств в сеть.

Примечание
В первом тесте представлена сеть, аналогичная иллюстрации. Нумерация устройств следующая:

1. Сервер
2. Магнитофон
3. Колонки
4. Обогреватель
5. Телевизор
6. Вентилятор
7. Спутниковая тарелка

Рассмотрим пример расчета промежуточных связей между вентилятором и другими устройствами. Вентилятор подключен напрямую к серверу, значит количество связей между вентилятором и сервером равно 1. Сервер, в свою очередь, подключен напрямую к обогревателю, телевизору и магнитофону. Таким образом количество связей между вентилятором и обогревателем равно 2, между вентилятором и телевизором равно 2, между вентилятором и магнитофоном так же равно 2. Количество связей между вентилятором и спутниковой тарелкой равно 3 (вентилятор-сервер, сервер-телевизор, телевизор-тарелка). Между вентилятором и колонками количество связей так же равно 3 (вентилятор-сервер, сервер-магнитофон, магнитофон-колонки). Следовательно, максимальное количество промежуточных связей от вентилятора до самых дальних от него устройств равно 3.

Формат входных данных
В первой строке дано целое число N – количество различных устройств ($2 \le N \le 10^5$).
Каждая из следующих $N-1$ строк содержит номер устройства id , к которому подключается очередное устройство.

Формат выходных данных

В единственной строке выведите номер устройства, являющегося центром полученной сети. Если таких устройств несколько, выведите их в порядке возрастания *id* .

Пример

Входные данные	**Выходные данные**
7 1 2 1 1 1 5	1

Решение:

Для решения данной задачи необходимо найти центр дерева. Обойдём граф из любой вершины, например, обходом в ширину. При обходе будем считать удалённость каждой вершины от вершины, с которой начинался обход. От самой удалённой вершины запустим обход ещё раз, так же подсчитывая расстояние от каждой вершины до стартовой, и запоминая из какой вершины был переход в текущую. Среди всех вершин выберем самую удалённую. Если расстояние четное, значит устройство, являющееся центром сети ровно одно. В противно случае таких устройств два. Для нахождения номеров вершин-центров пройдём из найденной самой удаленной вершины во время последнего обхода по сохраненным вершинам-предкам. Вершины, находящиеся в середине пути и будут являться ответом на задачу.

Код решения (C++)

```
#pragma comment(linker, "/STACK:64000000")
#define _CRT_SECURE_NO_WARNINGS
#include <iostream>
#include <string>
#include <vector>
#include <utility>

using namespace std;

int bfs(vector <vector <int> > &g, int v0, vector <int>
&dists, vector <int> &parents)
{
    vector <int> bfs_list;
    dists.assign(g.size(), -1);
    parents.assign(g.size(), -1);
    int head = 0;
    bfs_list.reserve(g.size());

    dists[v0] = 0;
    bfs_list.push_back(v0);
```

```
    while (head < bfs_list.size())
    {
        int v = bfs_list[head++];
        for (int i = 0; i < g[v].size(); ++i)
        {
            int v_to = g[v][i];
            if (dists[v_to] == -1)
            {
                dists[v_to] = dists[v] + 1;
                bfs_list.push_back(v_to);
                parents[v_to] = v;
            }
        }
    }

    int idx = v0;
    for (int i = 0; i < dists.size(); ++i)
    {
        if (dists[i] > dists[idx])
        {
            idx = i;
        }
    }

    return idx;
}

void swap(int &a, int &b)
{
    int c = a;
    a = b;
    b = c;
}

void print_center(vector <int> &parents, int from, int to)
{
    vector <int> path(parents.size());
    int cnt = 0;
    int v = to;
    while (v != from)
    {
        path[cnt++] = v;
        v = parents[v];
    }
    path[cnt++] = from;

    if (cnt & 1)
    {
        cout << path[cnt >> 1] + 1 << endl;
```

```
    }
    else
    {
        int v1 = path[cnt >> 1] + 1;
        int v2 = path[(cnt >> 1) - 1] + 1;
        if (v2 < v1)
        {
            swap(v1, v2);
        }
        cout << v1 << ' ' << v2 << endl;
    }
}

int main()
{
    int n;
    vector <vector <int> > g;
    vector <int> dists;
    vector <int> parents;
    cin >> n;
    g.assign(n, vector<int>(0));

    for (int i = 0; i < n - 1; ++i)
    {
        int v_cur = i + 1;
        int v;
        cin >> v;
        --v;
        g[v].push_back(v_cur);
        g[v_cur].push_back(v);
    }

    int v1 = bfs(g, 0, dists, parents);
    int v2 = bfs(g, v1, dists, parents);
    print_center(parents, v1, v2);
}
```