http://nvk26.lutsk.ua/cgi-bin/new-client?contest_id=59

school2018-1 … school2018-30

пароль 1

Розвязки

Проста задача?

Time limit: 1c

Memory limit: 64m

reeWorld любить садити Тарiка. Якось reeWorld задав Тарiку дуже цiкаву задача.

На здивування Тарiк зробив її достатньо швидко, тому reeWorld вирiшив знову все узагальнити.

Чи справитесь з новою задачею ви?

Вхідні дані:

В першому рядку задано 1<=T<=20 - кількість груп тестів.

В наступних T рядках знаходиться по одному числу 1<=A_i<=10^18.

Вихідні дані:

Для кожного заданого чила необхідно перевірити парність кількості дільників.

В кожному окремому рядку необхідно вивести Yes, якщо кількість дільників даного числа непарна, або No, якщо кількість дільників парна.

Приклад 1:

4

1

3

4

7

Yes

No

Yes

No

```
#include <fstream>
 #include <math.h>
 using namespace std;

 ifstream cin("input.txt");
 ofstream cout("output.txt");

 int main()
 {
        int t;
        cin>>t;

        while(t)
        {
                long long a, b;
                cin>>a;
                b = sqrt((double)a);

                if(b*b==a || (b+1)*(b+1)==a)
                {
                        cout<<"Yesn";
                }else
```

```
            {
                cout<<"Non";
            }

            t--;
        }
        return 0;
}
```

Точки на прямій

Timelimmit: 1c

MemoryLimit: 64m

Визначемо діаметр мультимножини(набору) точок на прямій як максимальна відстань між двома точками цієї мультимножини(набору).

Відповідно діаметр мультимножини, яка складається з однієї точки рівний 0.

Задано N точок. Яку мінімальну кількість точок потрібно видалити, щоб діаметр мультимножини точок які залишилися не перевищував d.

Вхідні дані:

В першому рядку задано два цілих числа N і d(1 ≤ n ≤ 100, 0 ≤ d ≤ 100) - кількість точок і обмеження на діаметр.

В другому рядку задано N чисел(1 ≤ Xi ≤ 100) - координати точок.

Вихідні дані:

Одне число - мінімальна кількість точок, які потрібно видалити.

Приклад 1:

3 1

2 1 4

1

Приклад 2:

3 0

7 7 7

0

Приклад 3:

6 3

1 3 4 6 9 10

3

Пояснення:

В першом тесті вигідно видалити точку з координатою 4. Новий діаметр буде рівний 2-1=1.

В другому тесті діаметр одразу рівний 0.

В третьому вигідно видалити точки з координатами 1, 9 і 10.

```
#include <fstream>
 #include <algorithm>
 using namespace std;

 ifstream cin("input.txt");
 ofstream cout("output.txt");

 int main()
 {
	int n, m[105], ma = -1000000, mi = 100000, i, tr = 10000, d, j;

	cin >> n >> d;
	for (i = 0; i < n; i++)
	{
		cin >> m[i];
		ma = max(ma, m[i]);
		mi = min(m[i], mi);
	}

	if (ma - mi <= d)
	{
		cout << 0;
		return 0;
	}

	sort(m, m + n);

	for (i = 0; i < n; i++)
	{
		for (j = i; j < n; j++)
		{
			if (m[j] - m[i] > d)
			{
				break;
			}
		}
		tr = min(tr, n - (j - i));
	}

	cout << tr;

	return 0;
 }
```

Ізіч

Time limmit: 1c

Memory Limit: 64m

Наші тести не готові. Скоро 3 тур. Але ж головна ціль авторів контеста - зробити ще одну задачу.

Дехто з ніком reeWorld хотів дати задачу на цетроїдну декомпозицію. Нехай задано граф, який є деревом.

Очевидно, що в такому графі всі вершини, крім листків, є шарнірами. Центроїдом даного графа буде такий шарнір, який розділяє даний граф на компонент-зв'язності розмірами в 2 і більше разів меншими ніж початковий.

Отже, постало Q запитань. Скільки серед заданих N чисел менших за A_i.

Вхідні дані:

В першому рядку знаходиться число 5<=N<=100000 - кількість заданих чисел.

В другому рядку знаходяться задані числа 0<=M_i<=1000000007.

В наступному рядку знаходиться кількість запитів 5<=Q<=100000.

В наступних Q рядках знаходяться по одному числу 0<=A_i<=1000000007.

Вихідні дані:

Для кожного і-ого запиту необхідно вивести кількість заданих чисел менших за A_i.

Приклад:

10

2 4 13 14 16 12 4 12 15 10

10

16

3

9

1

0

10

14

11

6

16

9

1

3

0

0

3

7

4

3

9

```
#include <fstream>
 #include <vector>
 #include <algorithm>
 #include <set>
 #include <time.h>
 using namespace std;

 ifstream cin("input.txt");
 ofstream cout("output.txt");

 //ofstream dat("dat1.txt");
 //ofstream ans("ans1.txt");

 #define N 100
 #define Q 10
 #define MOD 10007
 //#define MOD 1000000007

 vector<int> v;
 set <pair<int, int> > s;
 set <pair<int, int> >::iterator it;
```

```
int main()
{
	srand(time(NULL));

	if (1)
	{
		int n, i, a, q;

		cin >> n;
		for (i = 1; i <= n; i++)
		{
			cin >> a;
			v.push_back(a);
		}

		sort(v.begin(), v.end());

		for (i = 0; i < n; i++)
		{
			s.insert(make_pair(v[i], i));
		}

		cin >> q;

		for (i = 0; i < q; i++)
		{
			cin >> a;
			it = s.lower_bound(make_pair(a, -1));

			if (it == s.end())
			{
				cout << n << "n";
			}
			else
			{
				cout << it->second << "n";
			}
		}

		return 0;
	}
	else
	{
		/*long long i, j, a, b;
		dat << N<<"n";
		for (i = 0; i < N; i++)
```

```
            {
                a = rand()*rand()*rand()%MOD + rand() - rand();
                a %= MOD;
                if (a < 0) { a += MOD; }
                dat << a<<" ";
            }
            dat << "n" << Q << "n";

            for (i = 0; i < Q; i++)
            {
                a = rand()*rand()*rand() % MOD + rand() - rand();
                a %= MOD;
                if (a < 0) { a += MOD; }
                dat << a << "n";
            }*/
        }
}
```

Рядочки

Time limit: 1c

Memory limit: 64m

Вам задано рядок довжиною від 4 до 5000 символів, який складається лише з малих літер анлійського алфавіту.

Необхідно перевірити, чи можна розділити даний рядок на 4 непусті паліндроми.

Вхідні дані:

В єдиній стрічці знаходиться вхідний рядок.

Вихідні дані:

Виведіть Yes, якщо рядок можна розділити на 4 непусті паліндроми або No інакше.

Приклад 1:

abaaaa

Yes

Приклад 2:

abacaba

No

```
#include <fstream>

#include <algorithm>

#include <math.h>

#include <vector>

using namespace std;

ifstream cin("input.txt");

ofstream cout("output.txt");

vector <int> primeNumbers;

long long n, i, answer;

int *productComposition;
```

```
long long minAns;


bool isPrime(int number)


{


    int i;


    for (i = 2; i*i <= number; i++)


    {


        if (number%i == 0)


        {


            return false;


        }


    }


    return true;
```

```
}

void nextProducts(unsigned long long product, int productCompositionPosition, long long
numberOfDividers)

{

    for (; productComposition[productCompositionPosition - 1] >
productComposition[productCompositionPosition];)

    {

        productComposition[productCompositionPosition]++;

        numberOfDividers = numberOfDividers /
productComposition[productCompositionPosition] *
(productComposition[productCompositionPosition] + 1);

        product *= primeNumbers[productCompositionPosition - 1];

        if (product >= n)

        {
```

```
            break;

        }

        if (answer < numberOfDividers)

        {

            answer = numberOfDividers;

            minAns = product;

        }

        if (product*primeNumbers[productCompositionPosition] < n)

        {

            nextProducts(product, productCompositionPosition + 1, numberOfDividers);

        }

    }
```

```
    productComposition[productCompositionPosition] = 0;

}

int main()

{

    answer = 1;

    cin >> n;

    //lets find all necessary prime numbers

    unsigned long long timeN = 1;

    for (i = 2; timeN < n; i++)

    {

        if (isPrime(i))
```

```
        {

            timeN *= i;

            primeNumbers.push_back(i);

        }

    }

    // 1 = (1^BIG) * (2^0) * (3^0) * (5^0) * (7^0)...

    productComposition = new int[primeNumbers.size() + 1];

    productComposition[0] = log2(n) + 1;

    for (i = 1; i <= primeNumbers.size(); i++)

    {

        productComposition[i] = 0;

    }
```

```
    //main search

    nextProducts(1, 1, 1);

    //lets cout answer

    cout << answer << "n" << minAns;

    return 0;

}
```

Дільники

Time limmit: 1c

Memory Limit: 64m

Ви знаєте китайські теореми про остачі???

От і добре, в цій задачі вони не потрібні.

В цій задачі потрібно знайти число, яке було би менше або рівне за дане і мали би при цьому максимальну кількість дільників.

Вхідні дані:

Число 2<=N<=10^17

Вихідні дані:

В першому рядку неохідно вивести максимально можливу кількість дільників, яку може мати число менше рівне за N.

в другому рядку потрібно вивести число, яке би мало відповідну кількість дільників і було менше за N. Якщо таких чисел декілька, то виведіть найменше з них.

Приклад 1:

10

4

6

Приклад 2:

1000000

240

720720

Пояснення

У другому прикладі є 5 чисел менших за 10^6, які мають 240 дільників: 720720, 831600, 942480, 982800, 997920.

Примітка

Якби N=10^60 то відповідь була би:

521838526464

955140758164680860628432565596792358941635407885397923872000

```
#include <fstream>
 #include <algorithm>
 #include <math.h>
 #include <vector>
 using namespace std;

 ifstream cin("input.txt");
 ofstream cout("output.txt");

 vector <int> primeNumbers;
 long long n, i, answer;
 int *productComposition;
 long long minAns;

 bool isPrime(int number)
 {
        int i;
        for (i = 2; i*i <= number; i++)
        {
                if (number%i == 0)
                {
                        return false;
                }
        }
        return true;
 }

 void nextProducts(unsigned long long product, int productCompositionPosition, long long
numberOfDividers)
 {

        for (; productComposition[productCompositionPosition - 1] >
productComposition[productCompositionPosition];)
        {
                productComposition[productCompositionPosition]++;
                numberOfDividers = numberOfDividers /
productComposition[productCompositionPosition] *
(productComposition[productCompositionPosition] + 1);
                product *= primeNumbers[productCompositionPosition - 1];
                if (product >= n)
                {
                        break;
                }
```

```
            if (answer < numberOfDividers)
            {
                answer = numberOfDividers;

                minAns = product;
            }
            if (answer == numberOfDividers)
            {
                if (minAns>product)
                {
                    minAns = product;
                }
            }

            if (product*primeNumbers[productCompositionPosition] < n)
            {
                nextProducts(product, productCompositionPosition + 1, numberOfDividers);
            }
        }
        productComposition[productCompositionPosition] = 0;
}

int main()
{
        answer = 1;
        cin >> n;

        //lets find all necessary prime numbers
        unsigned long long timeN = 1;
        for (i = 2; timeN < n; i++)
        {
            if (isPrime(i))
            {
                timeN *= i;
                primeNumbers.push_back(i);
            }
        }

        // 1 = (1^BIG) * (2^0) * (3^0) * (5^0) * (7^0)...
        productComposition = new int[primeNumbers.size() + 1];
        productComposition[0] = log2(n) + 1;
        for (i = 1; i <= primeNumbers.size(); i++)
        {
            productComposition[i] = 0;
        }
```

```
	//main search
	nextProducts(1, 1, 1);

	//lets cout answer
	cout << answer << "n" << minAns;

	return 0;
}
```

Гроб

Time limmit: 1c

Memory Limit: 64m

reeWorld вирішив що 5 ізічів на один контест вже і так достатньо, тому добавив ще один.

Тарік зіткнувся з проблемою недостачі грошей, а тому покинув програмування і подався вкладати плитку.

У нього є безліч плиток Г-подібного троміно. Йому потрібно замостити кімнату розмірov 4*3
N
.

Скількома способами він може це зробити?

Найвідовіші вчителі математики в Волинській області зіткнулися з проблемою розв'язання даної задачі...

Проте, ви то точно вмієте виводити формулу в послідовності :) gggl

Вхідні дані:

Одне число 1<=N<=10^18

Вихідні дані:

Кількість можливих замощень кімнати.

Оскільки відповідь може бути достатньо великою, потрібно вивести її по модулю 100000 0007

```
#include <fstream>
 using namespace std;

 ifstream cin("input.txt");
 ofstream cout("output.txt");

 #define MOD 1000000007

 struct mat
 {
        long long a[3][3];
 };

 mat mult(mat a, mat b)
```

```
{
	mat c;
	long long i, j, l;
	for(i=0;i<3;i++)
	{
		for(j=0;j<3;j++)
		{
			c.a[i][j]=0;
			for(l=0;l<3;l++)
			{
				c.a[i][j]+=a.a[i][l]*b.a[l][j];
				c.a[i][j]%=MOD;
			}
		}
	}

	return c;
}

mat sqr(mat a)
{
	return mult(a, a);
}

mat binpow(mat a, long long n)
{
	if(n==1)
	{
		return a;
	}else if(n%2==0)
	{
		return sqr(binpow(a, n/2));
	}else
	{
		return mult(sqr(binpow(a, n/2)), a);
	}
}

int main()
{
	long long n, i, j;

	cin >> n;

	if(n==1)
	{
```

```
        cout<<4;
        return 0;
    }else if(n==2)
    {
        cout<<18;
        return 0;
    }else if(n==3)
    {
        cout<<88;
        return 0;
    }

    mat m;

    m.a[0][0] = 0; m.a[0][1] = 0; m.a[0][2] = -4;
    m.a[1][0] = 1; m.a[1][1] = 0; m.a[1][2] = -22;
    m.a[2][0] = 0; m.a[2][1] = 1; m.a[2][2] = 10;

    mat res = binpow(m, n-3);

    long long result = res.a[0][2]*4 + res.a[1][2]*18 + res.a[2][2]*88;
    result%=MOD;

    cout<<result;

    return 0;
}
```