

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего  
образования

**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ  
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г. ЧЕРНЫШЕВСКОГО»**

Кафедра теоретических основ  
компьютерной безопасности и  
криптографии

**Построение больших простых чисел с помощью теоремы Поклингтона**

**ЛАБОРАТОРНАЯ РАБОТА**

студента 4 курса 431 группы

специальности 10.05.01 Компьютерная безопасность

факультета компьютерных наук и информационных технологий

Серебрякова Алексея Владимировича

Научный руководитель

доцент, к. п. н.

\_\_\_\_\_  
подпись, дата

А. С. Гераськин

Саратов 2022

## Описание алгоритма

Критерий Поклингтона. Пусть  $n$  – натуральное число. Пусть число  $n - 1$  имеет простой делитель  $q$ , причем  $q > \sqrt{n} - 1$ . Если найдется такое целое число  $a$ , что выполняются следующие два условия:

1.  $a^{n-1} \equiv 1 \pmod{n}$
2. Числа  $n$  и  $a^{(n-1)/q} - 1$  – взаимно просты, то  $n$  – простое число.

## Код программы

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

using namespace std;

vector<int> prime_dividers;
vector<int> prime_dividers_set;

int validated_input()
{
    int s = 0;
    while (!(cin >> s))
    {
        cin.clear();
        cin.ignore(numeric_limits<streamsize>::max(), '\n');
        printf("! Неверный ввод. Повторите ввод, начиная с первого неверного элемента.\n");
    }

    return s;
}

void fact(int n)
{
    prime_dividers = {};

    int tmp(n);

    for (auto cnt = 2; cnt * cnt <= tmp; cnt++)
    {
        if (tmp % cnt == 0)
            prime_dividers.push_back(cnt);

        while (tmp % cnt == 0)
            tmp = tmp / cnt;
    }

    if (tmp > 1)
        prime_dividers.push_back(tmp);

    // printf("\nДелители n - 1 = %d", n - 1);
    // for(auto e: prime_dividers)
    //     printf(" [%d]", e);
}

void fact_set(int n)
```

```

{
    prime_dividers_set = {};

    int tmp(n);

    for (auto cnt = 2; cnt * cnt <= tmp; cnt++)
    {
        if (tmp % cnt == 0)
            prime_dividers_set.push_back(cnt);

        while (tmp % cnt == 0)
            tmp = tmp / cnt;
    }

    if (tmp > 1)
        prime_dividers_set.push_back(tmp);

    // printf("\nДелители n - 1 = %d", n - 1);
    // for(auto e: prime_dividers_set)
    //     printf(" [%d]", e);
}

int fastPow(int num, int deg)
{
    int result = 1;

    while (deg)
    {
        if (deg % 2 == 0)
        {
            deg /= 2;
            num *= num;
        }
        else
        {
            deg--;
            result *= num;
        }
    }

    return result;
}

bool pockTest(int a, int n, int q){
    if (fastPow(a, n-1) % n != 1)
        return false;

    fact_set(fastPow(a, (n-1)/q)-1);

    for(auto e: prime_dividers)
        for(auto e_set: prime_dividers_set)
            if (e == e_set) return false;

    return true;
}

pair<int, int> genPrime(int n)
{
    int a (n-1);

```

```

n = fastPow(2, n) - 1;

fact(n-1);

int q = prime_dividers[0];

if (q > sqrt(n) - 1)
    while (!pockTest(a, n, q) && a>1)
        a--;
return make_pair(a, n);
}

int main()
{
    setlocale(0, "");

    int n;
    pair<int, int> res;

    // printf("\nВведите число n:\n ");
    // n = validated_input();

    n = 3;
    res = genPrime(n);
    printf("\n\nПростое число для n = %3d: %7d | Свидетель: %4d\n\n",
        n, res.second, res.first);

    n = 5;
    res = genPrime(n);
    printf("\n\nПростое число для n = %3d: %7d | Свидетель: %4d\n\n",
        n, res.second, res.first);

    n = 7;
    res = genPrime(n);
    printf("\n\nПростое число для n = %3d: %7d | Свидетель: %4d\n\n",
        n, res.second, res.first);


    n = 13;
    res = genPrime(n);
    printf("\n\nПростое число для n = %3d: %7d | Свидетель: %4d\n\n",
        n, res.second, res.first);

    n = 17;
    res = genPrime(n);
    printf("\n\nПростое число для n = %3d: %7d | Свидетель: %4d\n\n",
        n, res.second, res.first);

    return 0;
}

```

Пример запуска программы

●  Executing task: /bin/bash -c ./build/Debug/outDebug


Простое число для n = 3: 7 | Свидетель: 2

Простое число для n = 5: 31 | Свидетель: 4

Простое число для n = 7: 127 | Свидетель: 6

Простое число для n = 13: 8191 | Свидетель: 12

Простое число для n = 17: 131071 | Свидетель: 16

 Terminal will be reused by tasks, press any key to close it.