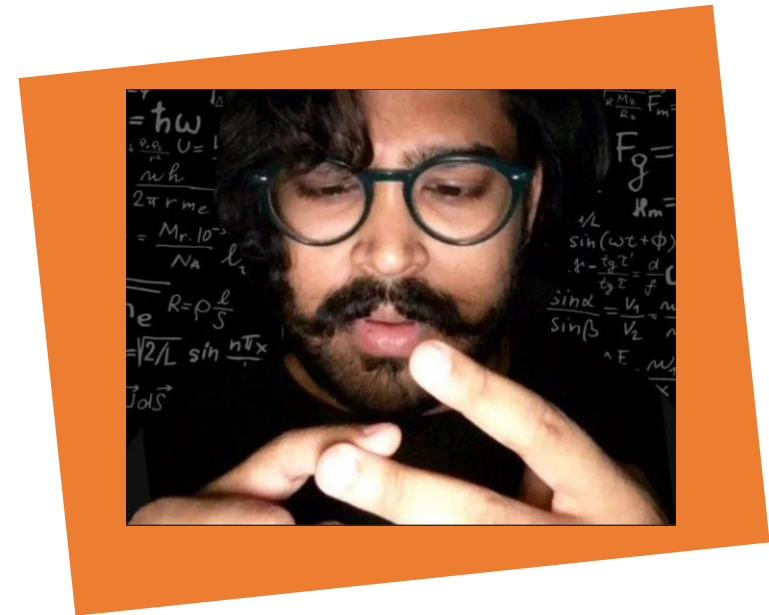
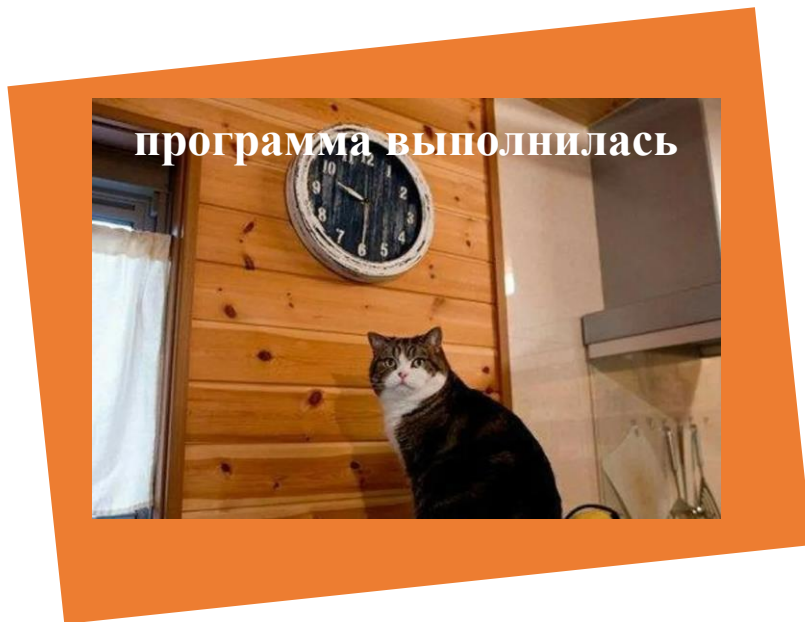


Введение в теорию сложности алгоритмов

10 класс

Как определить какой алгоритм эффективнее?

? Измерять время выполнения?



? Считать количество операций?



Что не так с подсчетом времени выполнения?



Один и тот же алгоритм может выполняться за **разное** время **даже на одном компьютере** от запуска к запуску, не говоря о разных компьютерах с разными процессорами.

6.292 с

```
#include <iostream>
#include <cmath>
#include <ctime>
using namespace std;

int main() {
    for (int i = -1000; i < 1000; i++) {
        for (int j = -1000; j < 1000; j++) {
            for (int k = -1000; k < 1000; k++) {
                if (i * k * j == 2 * (i + k + j)) {
                    cout << i << " " << j << " " << k << " " << endl;
                }
            }
        }
    }

    cout << "runtime = " << clock() / 1000.0 << endl;
}
```

Консоль отладки Microsoft Visual Studio

```
988 0 -988
989 -989 0
989 0 -989
990 -990 0
990 0 -990
991 -991 0
991 0 -991
992 -992 0
992 0 -992
993 -993 0
993 0 -993
994 -994 0
994 0 -994
995 -995 0
995 0 -995
996 -996 0
996 0 -996
997 -997 0
997 0 -997
998 -998 0
998 0 -998
999 -999 0
runtime = 6.292
```

6.284 с

```
#include <iostream>
#include <cmath>
#include <ctime>
using namespace std;

int main() {
    for (int i = -1000; i < 1000; i++) {
        for (int j = -1000; j < 1000; j++) {
            for (int k = -1000; k < 1000; k++) {
                if (i * k * j == 2 * (i + k + j)) {
                    cout << i << " " << j << " " << k << " " << endl;
                }
            }
        }
    }

    cout << "runtime = " << clock() / 1000.0 << endl;
}
```

Консоль отладки Microsoft Visual Studio

```
988 0 -988
989 -989 0
989 0 -989
990 -990 0
990 0 -990
991 -991 0
991 0 -991
992 -992 0
992 0 -992
993 -993 0
993 0 -993
994 -994 0
994 0 -994
995 -995 0
995 0 -995
996 -996 0
996 0 -996
997 -997 0
997 0 -997
998 -998 0
998 0 -998
999 -999 0
runtime = 6.284
```



Что не так с подсчетом времени выполнения?



Один и тот же алгоритм в разных языках программирования выполняется за разное время, иногда за СУЩЕСТВЕННО разное время.

Python: ~ 744 с ~ 12.5 мин

```
test.py - C:/Users/Evgeny/Desktop/test.py (3.11.5)
File Edit Format Run Options Window Help
import time

tm = time.time()
for i in range(-1000, 1001):
    for j in range(-1000, 1001):
        for k in range(-1000, 1001):
            if (i * j * k == 2 * (i + j + k)):
                print(i, j, k)
print(tm - time.time())

993 0 -993
994 -994 0
994 0 -994
995 -995 0
995 0 -995
996 -996 0
996 0 -996
997 -997 0
997 0 -997
998 -998 0
998 0 -998
999 -999 0
999 0 -999
1000 -1000 0
1000 0 -1000
-744.0940284729004
>>>
```

C++: ~ 6.3 с

```
#include <iostream>
#include <cmath>
#include <ctime>
using namespace std;

int main() {
    for (int i = -1000; i < 1000; i++) {
        for (int j = -1000; j < 1000; j++) {
            for (int k = -1000; k < 1000; k++) {
                if (i * k * j == 2 * (i + k + j))
                    cout << i << " " << j << " " << k << " ";
            }
        }
    }
    cout << "runtime = " << clock() / 1000.0 << endl;
}
```

Параметры сборки, на которой проводились тесты:

Процессор: 12th Gen Intel(R) Core(TM) i7-12700KF (20 CPUs), ~3.6GHz
Память: 32768MB RAM



Подсчет количества операций

Договоримся, что считать будем следующие операции в программе, остальное считаем выполняющимся мгновенно:

- Присваивание значения переменной
- Сравнение двух значений
- Инкрементация значения
- Основные арифметические операции: сложение, умножение и т.д.





Посчитаем количество операций для конкретного алгоритма

```
int main() {  
    int n, a;  
    int m = 0;  
    cin >> n;  
    for (int i = 0; i < n; i++) {  
        cin >> a;  
        if (a > m) {  
            m = a;  
        }  
    }  
}
```

Присваивание значения переменной

Сравнение двух значений

Инкрементация значения

Основные арифметические операции: сложение, умножение и т.д.



Наилучший и наихудший случаи

Подумайте, какой из данных случаев является наихудшим, а какой наилучшим?

```
int main() {  
    int n, a;  
    int m = 0;  
    cin >> n;  
    for (int i = 0; i < n; i++) {  
        cin >> a;  
        if (a > m) {  
            m = a;  
        }  
    }  
}
```

4

1

2

3

4

4

4

3

2

1

Задание 1. Подсчитайте количество операций для данных алгоритмов. Определите наилучший и наихудший случай.

а)

```
int main() {  
    for (int i = 0; i < 100; i++) {  
        cout << i;  
    }  
}
```

б)

```
int N;  
cin >> N;  
int s = 0;  
for (int i = 0; i < N; i++) {  
    for (int j = 0; j < N; j++) {  
        s = s + i * j;  
    }  
}
```


Задание 2. Определить асимптотическую сложность алгоритма с заданным в виде функции количеством операций.

а) $f(n) = 5n + 12$

б) $f(n) = 109$

в) $f(n) = n^2 + 3n + 112$

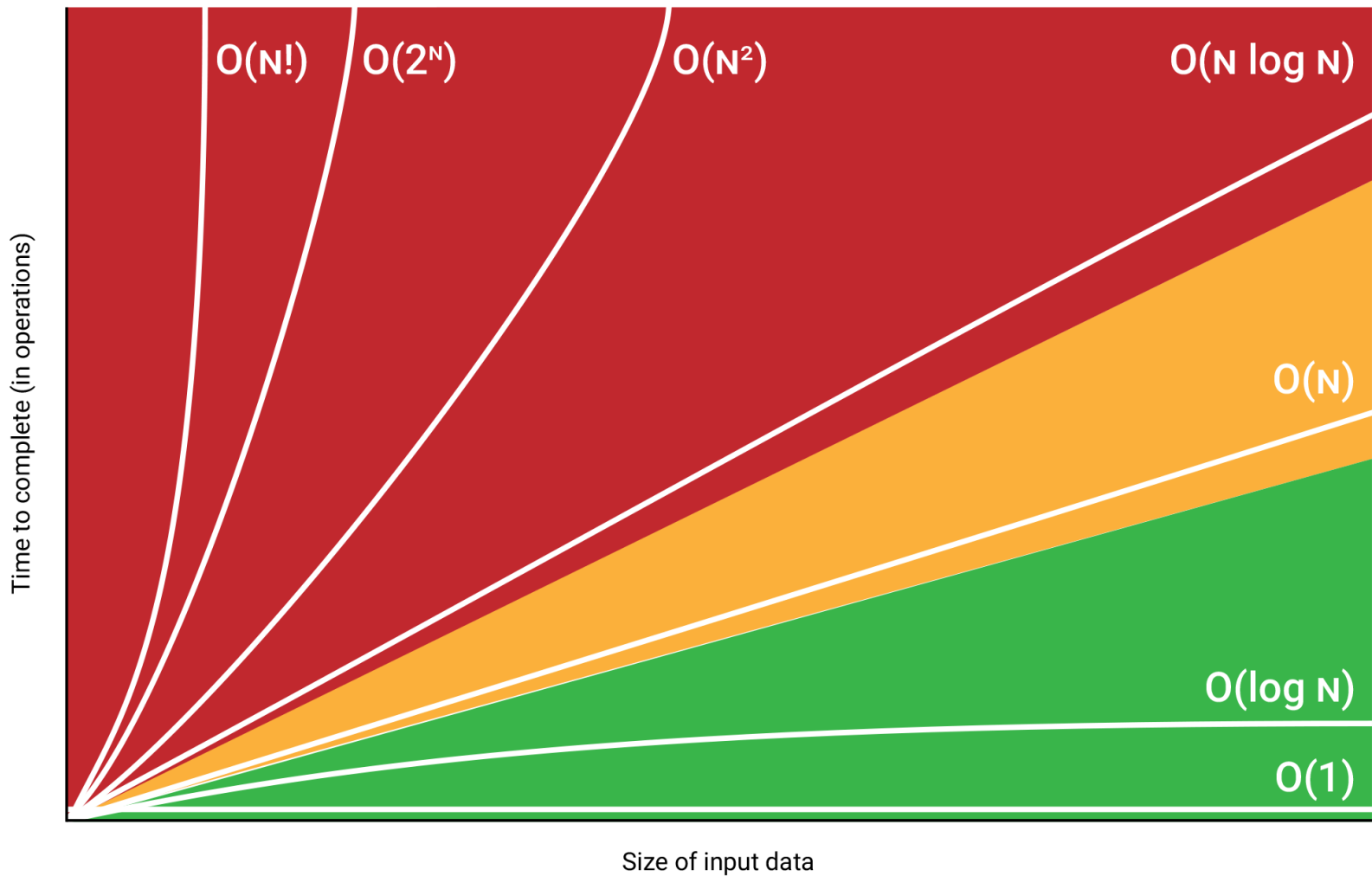
г) $f(n) = n^3 + 1999n + 1337$

д) $f(n) = n + \textit{sqrt}(n)$



Big O нотация

Факториальная	$O(N!)$	
Экспоненциальная	$O(2^N)$	
Квадратичная	$O(N^2)$	
Линейно-логарифмическая	$O(N \cdot \log N)$	
Линейная	$O(N)$	
Сублинейная	$O(\sqrt{N})$	
Логарифмическая	$O(\log N)$	
Константная	$O(1)$	





Примеры алгоритмов разной сложности

Константная:

```
int main() {  
    for (int i = 0; i < 100; i++) {  
        cout << i;  
    }  
}
```

Сублинейная:

```
int N;  
cin >> N;  
int s = 0;  
for (int i = 0; i * i < N; i++) {  
    cout << i;  
}
```



Примеры алгоритмов разной сложности

Линейная:

```
int N;  
cin >> N;  
int s = 0;  
for (int i = 0; i < N; i++) {  
    cout << i;  
}
```

Квадратичная:

```
int N;  
cin >> N;  
int s = 0;  
for (int i = 0; i < N; i++) {  
    for (int j = 0; j < N; j++) {  
        s = s + i * j;  
    }  
}
```

Повторим изученное

#ОЦЕНКА СЛОЖНОСТИ АЛГОРИТМОВ

Домашнее задание

1. Выучить виды сложностей алгоритмов в нотации Big O
2. Знать разобранные на уроке примеры под различные виды сложности алгоритмов