

# Кодирование информации

## § 11. Кодирование текстов

# Равномерное кодирование

1) выбрали алфавит:

{А, Б, В, ..., Я, ...}

всего  $M$  символов (мощность)



Сколько битов  
нужно?

2) длина кода  $i$  битов, так что:  $2^i \geq M$

3) составили кодовую таблицу:

<b>А</b>	<b>Б</b>	<b>В</b>	...	<b>Я</b>	...
<b>0</b>	<b>1</b>	<b>2</b>	...	<b>32</b>	...

В ДВОИЧНОМ КОДЕ

<b>000000</b>	<b>000001</b>	<b>000010</b>	...	<b>100000</b>	...
---------------	---------------	---------------	-----	---------------	-----

# Шрифты

Файл `vasya.txt` («ТОЛЬКО ТЕКСТ»):

01000001	01000010	01000011
----------	----------	----------

65

66

67

КОДЫ  
СИМВОЛОВ



Какие это символы?

Кодовая таблица:

...	<b>A</b>	<b>B</b>	<b>C</b>	...
...	65	66	67	...



Где взять изображение?

в файле его нет!

Шрифтовой файл (`.ttf`, `.otf`):

65 → **A**66 → **B**67 → **C**

Times New Roman



# Кодировка (кодированная таблица) ASCII

**ASCII** = American Standard Code for Information Interchange

Всего **128** символов, коды с **0** до **127**



Сколько бит нужно?

7-битная кодировка

48 – 57 цифры **0..9**

65 – 90 заглавные латинские буквы **A-Z**

97 – 122 строчные латинские буквы **a-z**

знаки препинания: . , : ; ! ?

специальные знаки: + - \* / ( ) { } [ ]



Международный стандарт!

# Однobaйтные кодировки

Добавим 1 бит до 8 бит на символ.



Сколько символов можно закодировать?

Всего **256** символов, коды с **0** до **255**

0	1	127	128	254	255
коды ASCII			расширение (национальный алфавит)		

## Кодовая страница (расширенная таблица ASCII)

для русского языка:

**Windows-1251** – для системы *Windows* (Интернет)

**KOI8-R** – для системы *UNIX* (Интернет)

**CP-866** – альтернативная кодировка (для системы *MS DOS*)

**MacCyrillic** – для компьютеров фирмы *Apple*

# Однobaйтные кодировки

---



- небольшой размер файла
- удобно обрабатывать программистам



- МОЖНО ИСПОЛЬЗОВАТЬ ТОЛЬКО 256 СИМВОЛОВ

- неизвестно какая кодировка

Windows-1251

KOI8-R

Здравствуй, мир!  
ъДТБЧУФЧХК, НЙТ!

гДПЮБЯРБСИ, ЛХП!  
Здравствуй, мир!

# Кодировки UNICODE

нужно  
больше!

**Идея:** все символы в одну таблицу!

16 битов на символ  $\rightarrow 2^{16} = 65536$  символов

**Стандарт UNICODE:** место для 1 112 064 символов

Windows:

**UTF-16** 2 или 4 байта на символ

заполнены  
~ 136 тысяч (2017)

Linux и др.:

**UTF-8** от 1 до 4 байт на символ

более 80% сайтов



Символы ASCII имеют те же коды!



■ можно использовать много символов



- увеличивается объём файла
- сложнее обрабатывать, если переменное число байт на символ

# Информационный объём текста

При равномерном кодировании:

$$I = L \cdot i$$

Количество информации ←  $I$  → Место для 1 символа

Количество символов ←  $L$  →

←  $i$  →

**Задача.** Определите информационный объём сообщения

**ПРИВЕТ, МИР!**

при использовании 16-битной кодировки.

- 1)  $L = 12$  символов
- 2)  $i = 16$  бит = **2 байта**
- 3)  $I = 12 \cdot 16 = 192$  бита  
 $I = 12 \cdot 2 =$  **24 байта**



Считаем пробелы и знаки препинания!



# Информационный объём текста

**Задача.** Рассказ, набранный на компьютере, содержит 12 страниц, на каждой странице 48 строк, в каждой строке 64 символа. Определите информационный объём рассказа в Кбайтах в кодировке, в которой каждый символ кодируется 16 битами.

$$I = L \cdot i$$

$$3 \cdot 4 = 3 \cdot 2^2$$

$$3 \cdot 16 =$$

$$3 \cdot 2^4$$

$$2^6$$

1)  $L = 12 \cdot 48 \cdot 64 = 9 \cdot 2^{12}$  СИМВОЛОВ

2)  $i = 16$  бит = 2 байта

3)  $I = 9 \cdot 2^{12} \cdot 2 = 9 \cdot 2^{13}$

$$= \frac{9 \cdot 2^{13}}{2^{10}} = 72 \text{ Кбайт}$$

байтов



1 Кбайт =  $2^{10}$  байтов!

# Информационный объём текста

**Задача.** Информационный объём статьи **96 Кбайт.**

Сколько страниц займет статья, если на одной странице электронного документа помещается 32 строки по 64 символа, а каждый символ занимает 8 бит памяти?.

 $2^5$ 
 $2^6$ 

$$I = L \cdot i$$

$$1) L = x \cdot 32 \cdot 64 = x \cdot 2^{11}$$

$$2) i = 8 \text{ бит} = 1 \text{ байт}$$

$$3) I = x \cdot 2^{11} \cdot 1 = x \cdot 2^{11} \text{ байт} = 96 \text{ Кбайт}$$

$$4) x \cdot 2^{11} \text{ байт} = 96 \cdot 2^{10}$$

байт

$$x \cdot 2 = 96$$

$$x = 48 \text{ страниц}$$

$$1 \text{ Кбайт} = 2^{10} \text{ байт!}$$

# Информационный объём текста

**Задача.** Некоторый файл перекодировали из 8-битной кодировки в 16-битную, в результате его информационный объём увеличился на 40 Кбайт. Чему был равен информационный объём файла до перекодирования?

1) информационный объём увеличился в 2 раза

$$I_2 = 2 I_1$$

2) по условию

$$I_2 = I_1 + 40$$

3) тогда

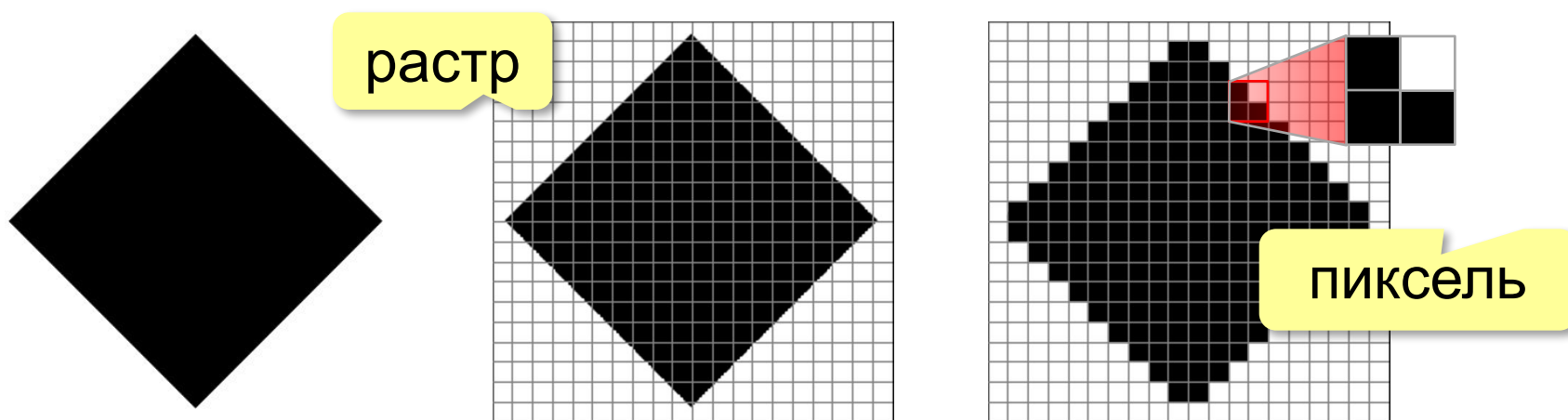
$$2I_1 = I_1 + 40$$

4)  $I = 40$  Кбайт

# Кодирование информации

## § 12. Кодирование рисунков: растровый метод

# Растровое кодирование



дискретизация



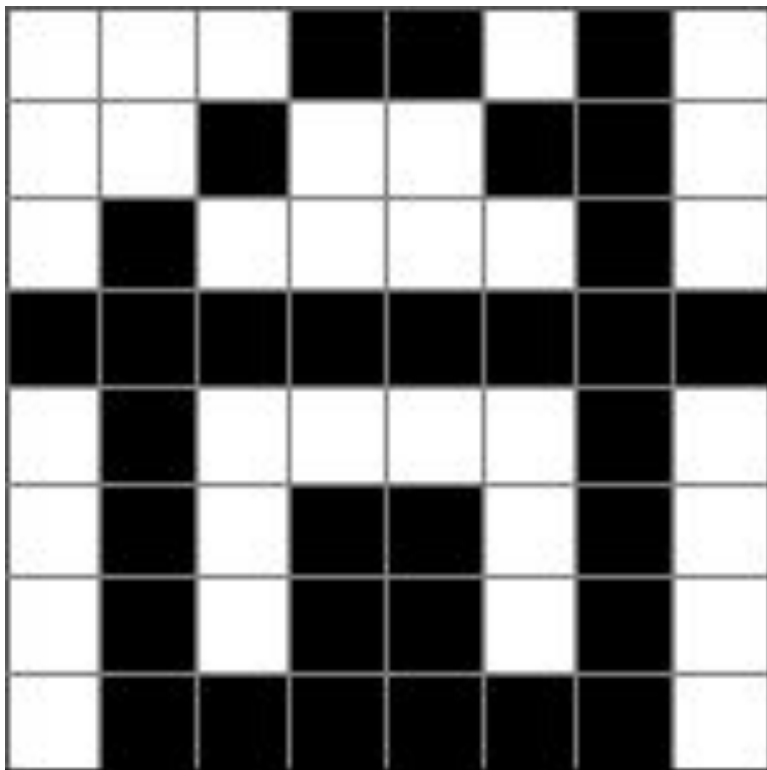
Рисунок искажается!



**Пиксель** – это наименьший элемент рисунка, для которого можно задать свой цвет.

**Растровое изображение** – это изображение, которое кодируется как множество пикселей.

# Растровое кодирование



1	1	1	0	0	1	0	1	E5
1	1	0	1	1	0	0	1	D9
1	0	1	1	1	1	0	1	BD
0	0	0	0	0	0	0	0	00
1	0	1	1	1	1	0	1	BD
1	0	1	0	0	1	0	1	A5
1	0	1	0	0	1	0	1	A5
1	0	0	0	0	0	0	1	81

**E5D9BD00BDA5A581**

# Разрешение

**Разрешение** – это количество пикселей, приходящихся на дюйм размера изображения.

*ppi* = *pixels per inch*, пикселей на дюйм

1 дюйм = 2,54 см



300 ppi

печать

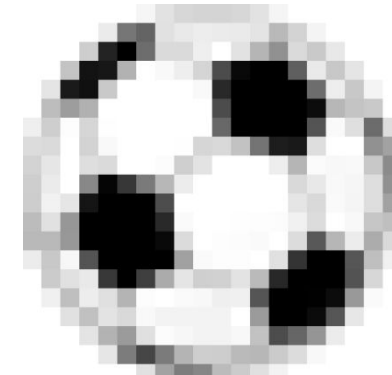


96 ppi

экран



48 ppi



24 ppi

## Разрешение

---

**Задача.** Какой размер в пикселях должен иметь закодированный рисунок с разрешением **300 ppi**, чтобы с него можно было сделать отпечаток размером **10×15 см**?

$$\text{высота} \quad \frac{10 \text{ см} \times 300 \text{ пикселей}}{2,54 \text{ см}} \approx \mathbf{1181 \text{ пиксель}}$$

$$\text{ширина} \quad \frac{15 \text{ см} \times 300 \text{ пикселей}}{2,54 \text{ см}} \approx \mathbf{1771 \text{ пиксель}}$$



# Кодирование цвета



00	11	11	11	11	11	11	11
00	11	11	11	11	11	11	11
00	01	01	01	01	01	01	01
00	01	01	01	01	01	01	01
00	10	10	10	10	10	10	10
00	10	10	10	10	10	10	10



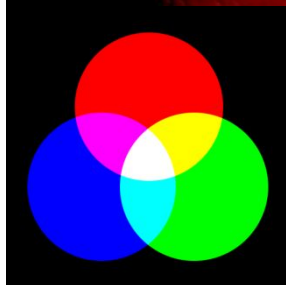
Как выводить на монитор цвет с кодом 00?



Как закодировать цвет в виде чисел?

# Цветовая модель RGB

Д. Максвелл, 1860



цвет = ( **R**, **G**, **B** )

*red*      *green*      *blue*

красный    зеленый    синий

0..255    0..255    0..255

■ (0, 0, 0)

■ (0, 255, 0)

□ (255, 255, 255)

■ (255, 255, 0)

■ (255, 0, 0)

■ (0, 0, 255)

■ (255, 150, 150)

■ (100, 0, 0)

■ (150, 150, 150)

■ (20, 20, 20)



Сколько разных цветов можно кодировать?

$256 \cdot 256 \cdot 256 = 16\ 777\ 216$  (*True Color*, «истинный цвет»)



RGB – цветовая модель для устройств, излучающих свет (мониторов)!

# Глубина цвета

**Глубина цвета** — это количество битов, используемое для кодирования цвета пикселя.



Сколько памяти нужно для хранения цвета 1 пикселя в режиме *True Color*?

**R** (0..255) 256 =  $2^8$  оттенков      8 битов = 1 байт

**R G B**: 24 бита = 3 байта

*True Color*  
(истинный цвет)

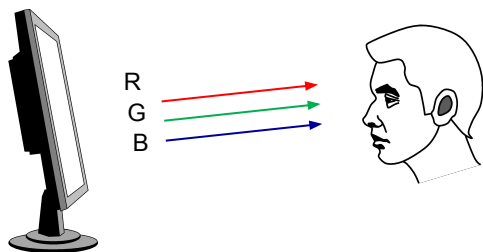
**Задача.** Определите размер файла, в котором закодирован растровый рисунок размером **20×30 пикселей** в режиме истинного цвета (*True Color*).

$20 \cdot 30 \cdot 3 \text{ байта} = \mathbf{1800}$

**байт**

# Кодирование цвета при печати (СМУК)

монитор



Белый – красный

Белый – зелёный

Белый – синий

= голубой

= пурпурный

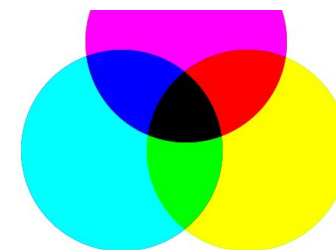
= желтый

печатный документ

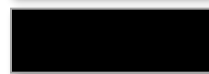
**C = Cyan**

**M = Magenta**

**Y = Yellow**



Модель CMY

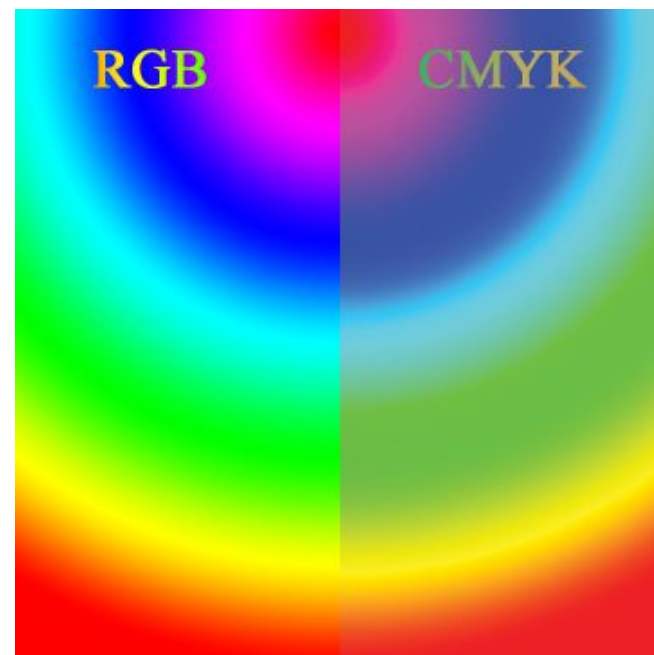
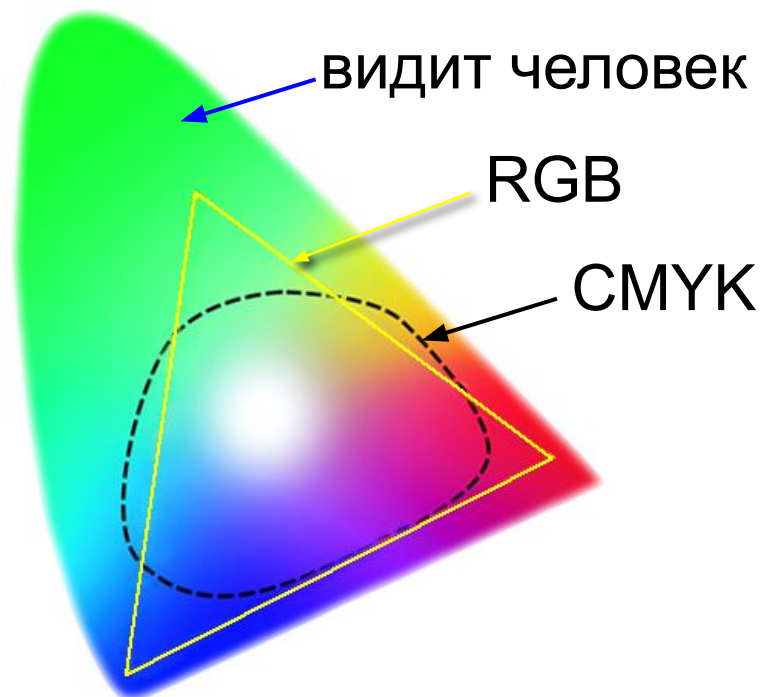


Модель CMYK: + **Key color**



- меньший расход краски и лучшее качество для чёрного и серого цветов

# RGB и CMYK



- не все цвета, которые показывает монитор (RGB), можно напечатать (CMYK)
- при переводе кода цвета из RGB в CMYK цвет искажается

**RGB(0,255,0)**  

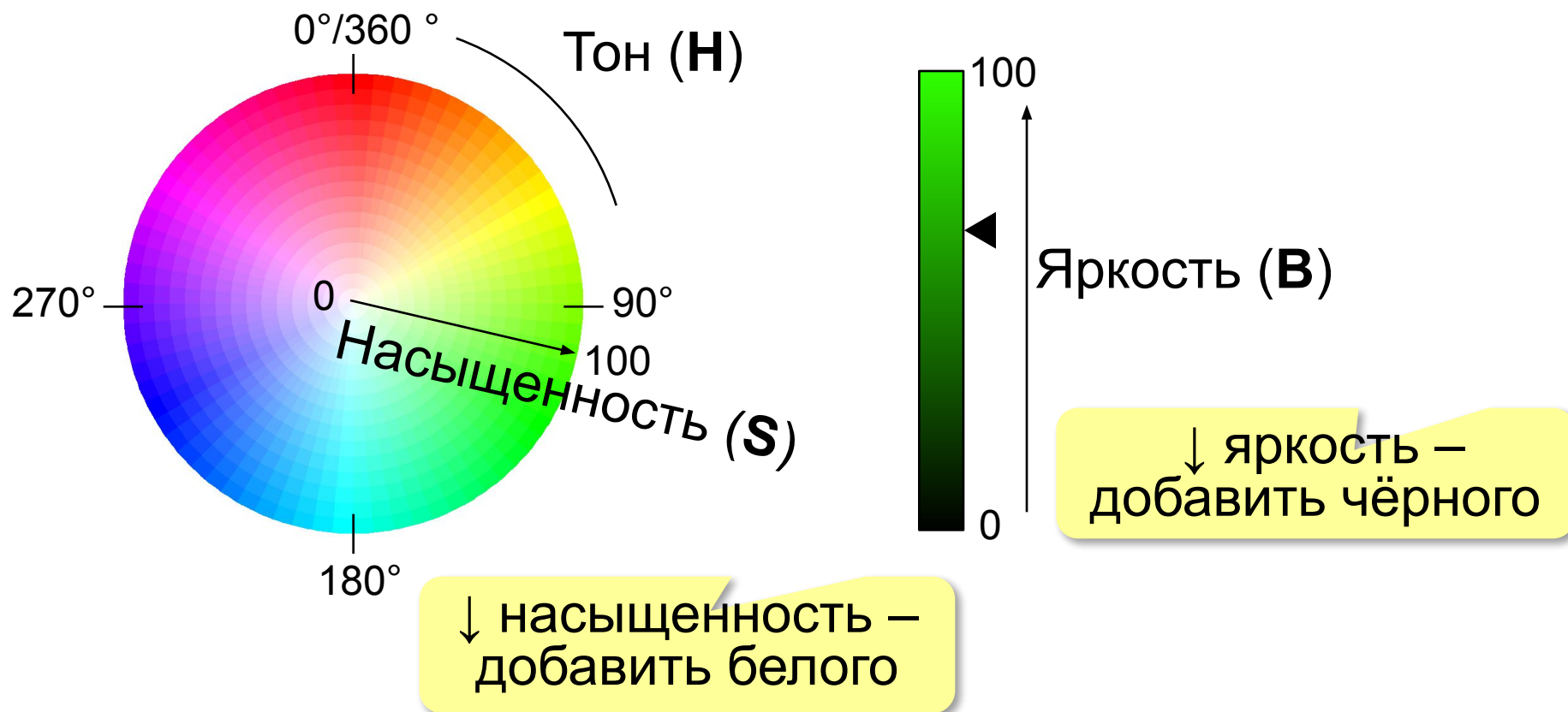
**CMYK(65,0,100,0)**  
 → **RGB(104,175,35)**

# Цветовая модель HSB (HSV)

**HSB** = *Hue* (тон, оттенок)

*Saturation* (насыщенность)

*Brightness* (яркость) или *Value* (величина)



# Кодирование с палитрой

---



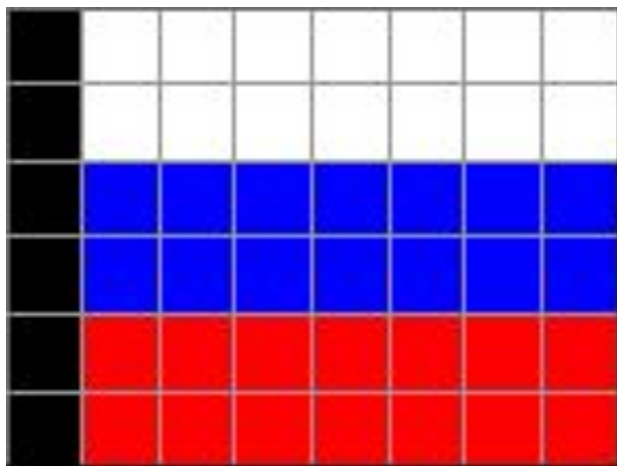
Как уменьшить размер файла?

- уменьшить разрешение
- уменьшить глубину цвета

снижается  
качество

**Цветовая палитра** – это таблица, в которой каждому цвету, заданному в виде составляющих в модели RGB, сопоставляется числовой код.

# Кодирование с палитрой



00	11	11	11	11	11	11	11
00	11	11	11	11	11	11	11
00	01	01	01	01	01	01	01
00	01	01	01	01	01	01	01
00	10	10	10	10	10	10	10
00	10	10	10	10	10	10	10

Палитра:

0	0	0	0	0	255	255	0	0	255	255	255
цвет $00_2$			цвет $01_2$			цвет $10_2$			цвет $11_2$		



Какая глубина цвета?

2 бита на пиксель



Сколько занимает палитра?

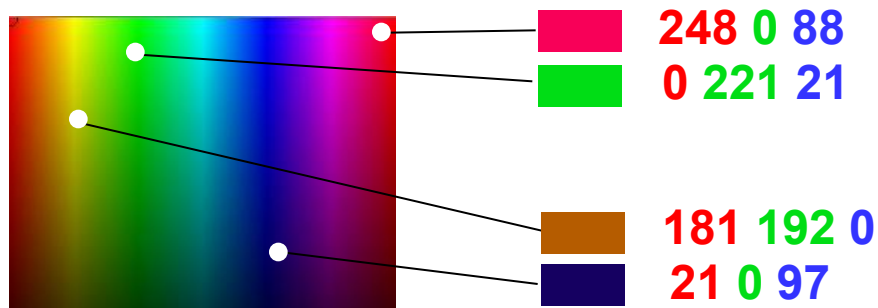
$3 \cdot 4 = 12$  байтов



# Кодирование с палитрой

Шаг 1. Выбрать количество цветов: 2, 4, ... 256.

Шаг 2. Выбрать 256 цветов из палитры:



Шаг 3. Составить палитру (каждому цвету – номер 0..255)  
палитра хранится в начале файла

0	1	45	254	255
248 0 88	0 221 21	148 35 115	181 192 0	21 0 97

Шаг 4. Код пикселя = номеру его цвета в палитре

1	254	45	14	...	12	23
---	-----	----	----	-----	----	----

## Кодирование с палитрой

**Задача.** Определите размер файла, в котором закодирован растровый рисунок размером  $20 \times 30$  пикселей с 16 цветной палитрой. Место для хранения палитры не учитывать.

- 1) количество пикселей  $20 \cdot 30 = 600$
- 2) 16 цветов  $\rightarrow$  4 бита на пиксель ( $16 = 2^4$ )
- 3)  $600 \cdot 4 = 2400$  бита = **300 байт**

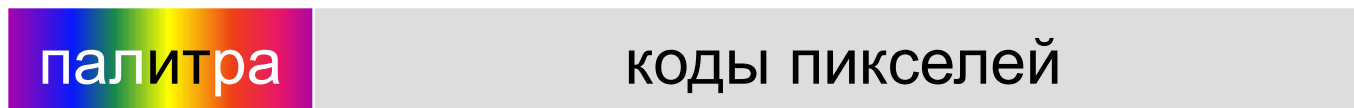


Сколько занимает палитра?

$$3 \cdot 16 = 48 \text{ байтов}$$

# Кодирование с палитрой

















Файл с палитрой:



3 байта на цвет



Количество цветов	Размер палитры (байтов)	Глубина цвета (битов на пиксель)
2	6	1
4	12	2
16	48	4
256	768	8

# Растровые рисунки: форматы файлов

Формат	True Color	Палитра	Прозрачность	Анимация
<b>BMP</b>				
<b>JPG</b>				
<b>GIF</b>				
<b>PNG</b>				

# Растровое кодирование: итоги

---

-  универсальный метод (можно закодировать любое изображение)
- единственный метод для кодирования и обработки размытых изображений, не имеющих чётких границ (фотографий)
-  **есть потеря информации** (почему?)
- при изменении размеров цвет и форма объектов на рисунке **искажаются**
- **размер файла** не зависит от сложности рисунка (а от чего зависит?)

# Кодирование информации

## § 13. Кодирование рисунков: другие методы

# Векторное кодирование

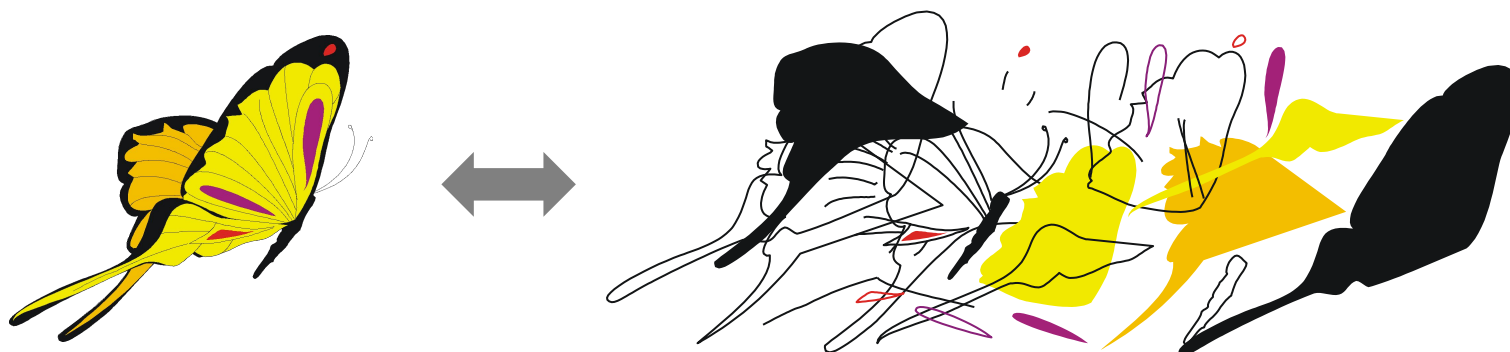
---

## Рисунки из геометрических фигур:

- отрезки, ломаные, прямоугольники
- окружности, эллипсы, дуги
- сглаженные линии (кривые Безье)

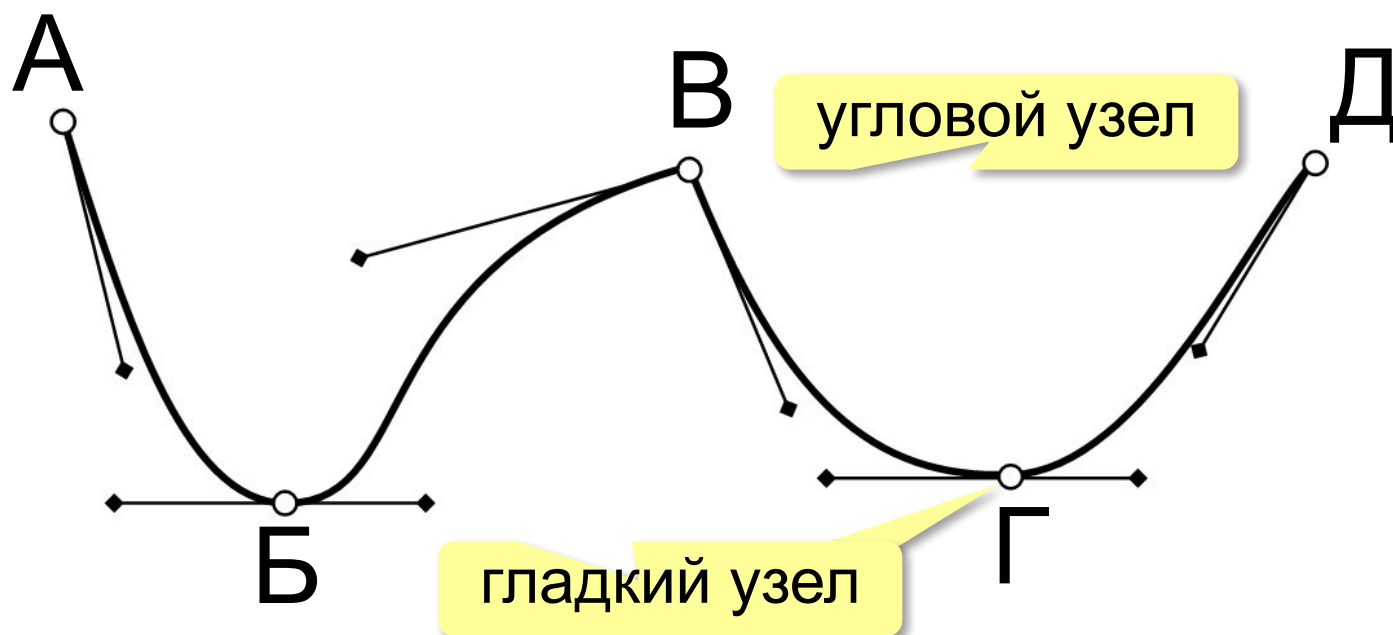
## Для каждой фигуры в памяти хранятся:

- размеры и координаты на рисунке
- цвет и стиль границы
- цвет и стиль заливки (для замкнутых фигур)



# Векторное кодирование

## Кривые Безье:



Хранятся координаты узлов и концов «рычагов»  
(3 точки для каждого узла, кривые 3-го порядка).



# Векторное кодирование (итоги)

---



- лучший способ для **чертежей, схем, карт**
- при кодировании **нет потери информации**
- при изменении размера **нет искажений**



A

растровый  
рисунок



A

векторный  
рисунок

- меньше **размер файла**, зависит от сложности рисунка



- не используют для **фотографий** и **размытых изображений**

# Векторное кодирование: форматы файлов

- **WMF, EMF** (*Windows Metafile*)
- **ODG** (формат *OpenOffice Draw*)
- **CDR** (программа *CorelDraw*)
- **AI** (программа *Adobe Illustrator*)
- **EPS** (для подготовки печатных изданий)
- **SVG** (*Scalable Vector Graphics*, масштабируемые векторные изображения)

для веб-страниц

# Векторные рисунки: SVG

<svg>

прямоугольник

размеры

```
<rect width="135" height="30"  
  x="0" y="10"  
  stroke-width="1" stroke="rgb(0,0,0)"  
  fill="rgb(255,255,255)" />
```

координаты

контур

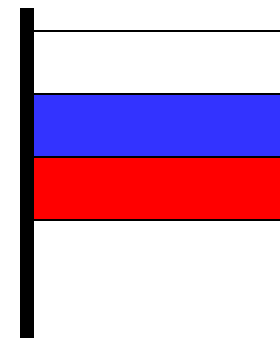
заливка

```
<rect width="135" height="30" x="0" y="40"  
  stroke-width="1" stroke="rgb(0,0,0)"  
  fill="rgb(0,0,255)" />
```

```
<rect width="135" height="30" x="0" y="70"  
  stroke-width="1" stroke="rgb(0,0,0)"  
  fill="rgb(255,0,0)" />
```

```
<line x1="0" y1="0"  
  x2="0" y2="150"  
  stroke-width="15" stroke="rgb(0,0,0)" />
```

</svg>

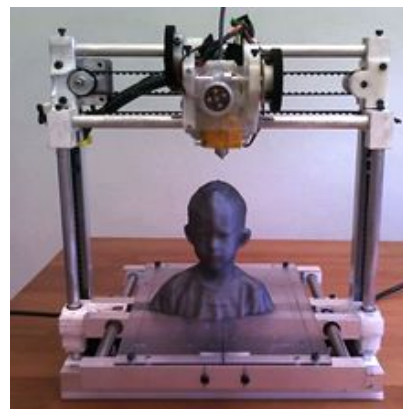
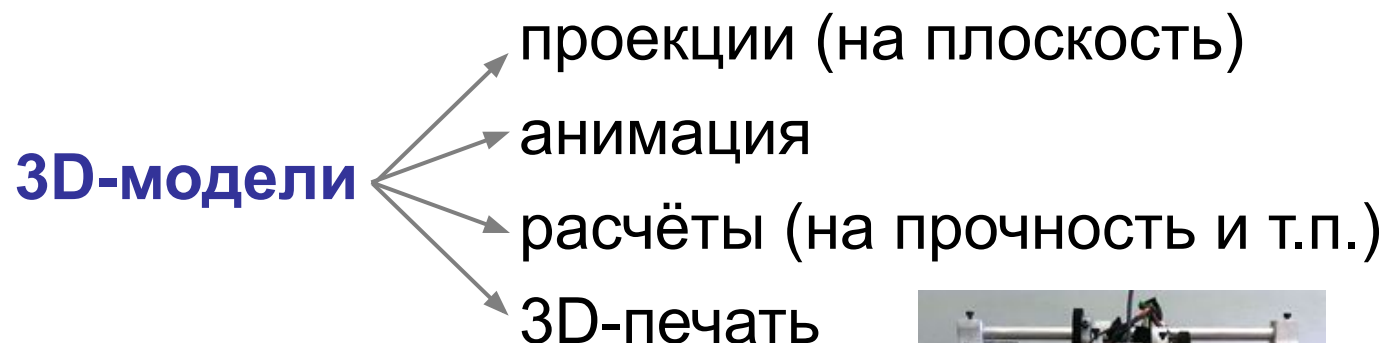


Это программа для браузера!

# 3D-графика

**Трёхмерная графика** (3D-графика) – это раздел компьютерной графики, который занимается созданием моделей и изображений *трёхмерных* объектов.

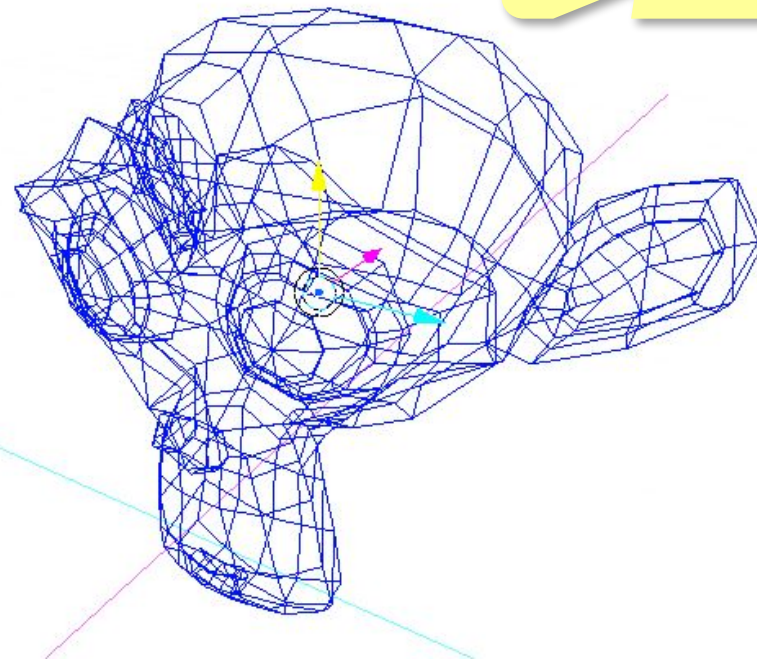
3D-модели: каждая точка имеет 3 координаты



# Построение каркаса (рёбер)

узлы  
(вершины)

рёбра



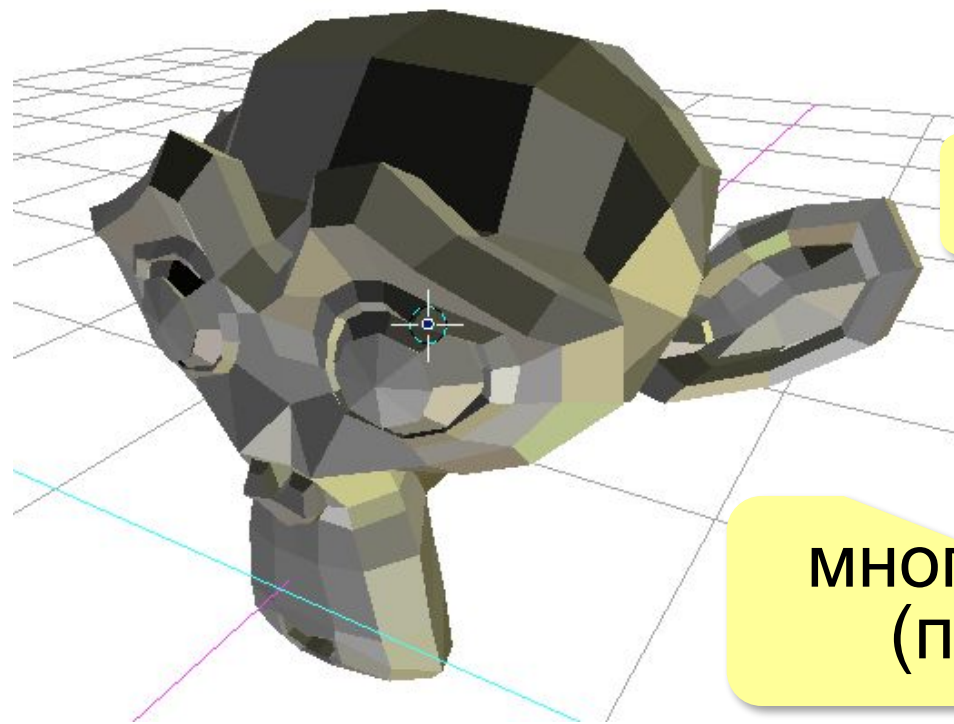
Хранятся координаты точек  $(x, y, z)$ !



~~Раскрасочная или векторная?~~

# Поверхность

---

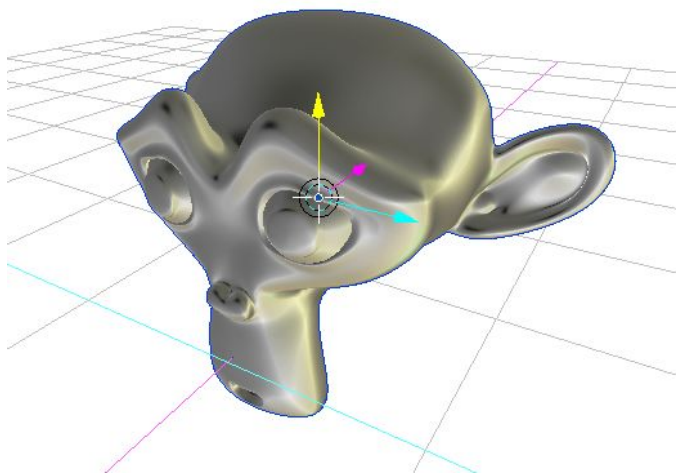


треугольники

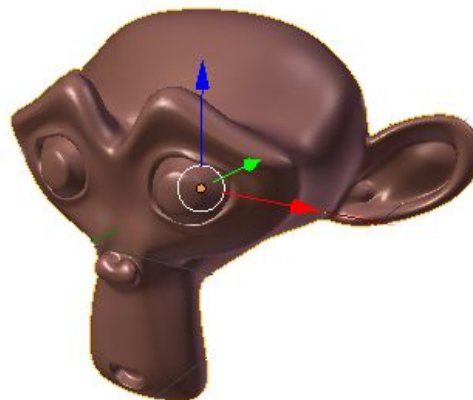
многоугольники  
(полигоны)

# Завершение модели

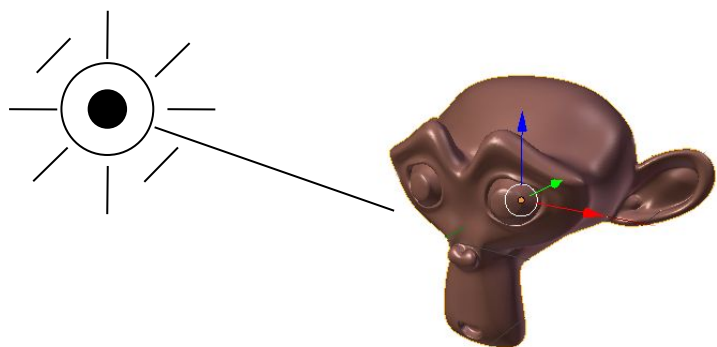
сглаживание



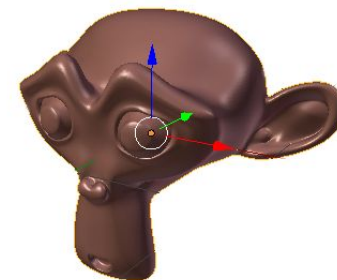
материал



установка света



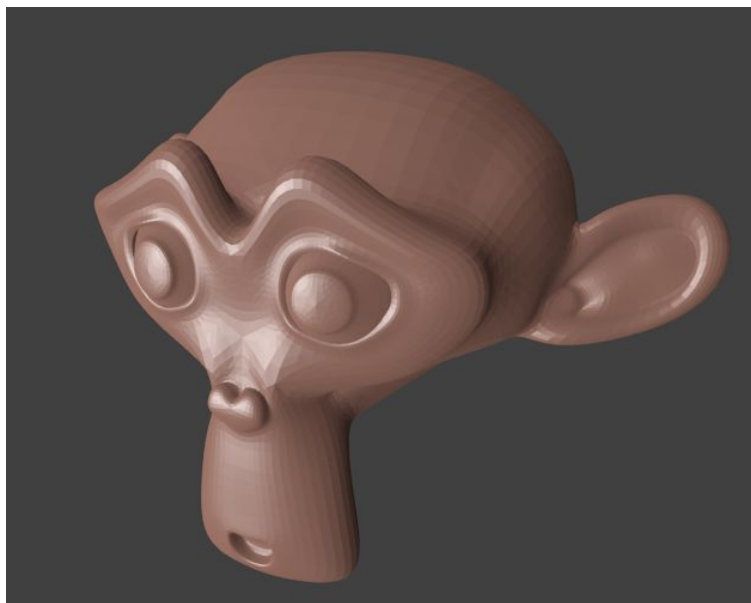
установка камеры



# Результат

---

рендеринг



3D-печать



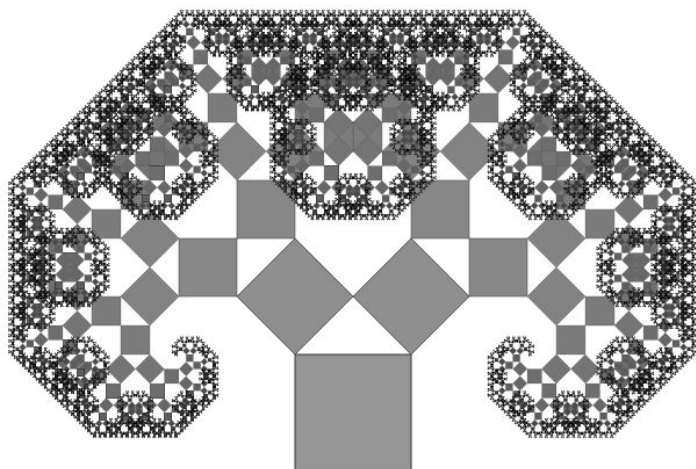
**Рендеринг** (визуализация) — построение двухмерного изображения по 3D-модели.



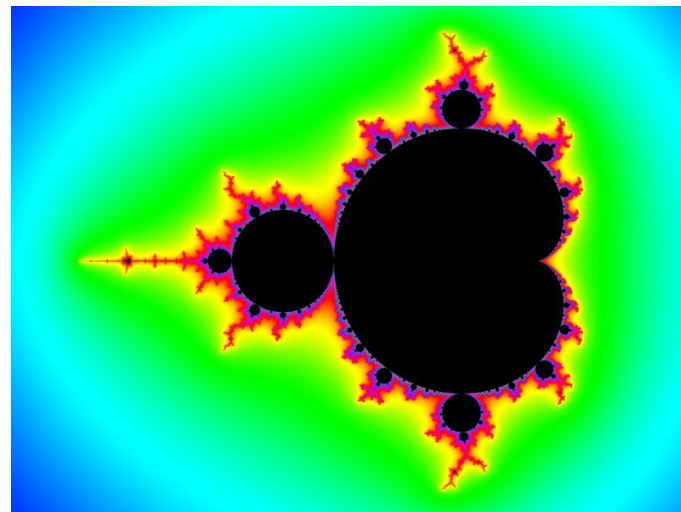
# Фрактальная графика

**Фрактал** — это фигура, обладающая *самоподобием*: основная фигура состоит из нескольких таких же, только меньшего размера.

## Пифагорово дерево



## Множество Мандельброта



Задаётся математической формулой + алгоритмом построения.

# Конец фильма

---

**ПОЛЯКОВ Константин Юрьевич**

д.т.н., учитель информатики

ГБОУ СОШ № 163, г. Санкт-Петербург

[kpolyakov@mail.ru](mailto:kpolyakov@mail.ru)

**ЕРЕМИН Евгений Александрович**

к.ф.-м.н., доцент кафедры мультимедийной

дидактики и ИТО ПГГПУ, г. Пермь

[eremin@pspu.ac.ru](mailto:eremin@pspu.ac.ru)

# Источники иллюстраций

---

1. <http://s1.iconbird.com>
2. <https://sandstorm.deviantart.com>
3. <http://http://compression.ru>
4. <http://ru.wikipedia.org>
5. <https://www.khanacademy.org>
6. <https://www.kns.ru>
7. <http://nix.ru>
8. <http://http://www.computer-services.ru>
9. <http://http://www.masterna4as.com>
10. <http://blendercontest.com>
11. <http://http://geeky-gadgets.com>
12. авторские материалы