16 с.с. (2, dec)

$0x1.8p1$ — $1.2e1 \to 12$ знаки после ,

2 с.с.

$0001.1000 \times 2^1 = 1.1 \times 2 = 11_2 = 3_{10}$

Сударь, ваш вопрос не является актуальным
оставьте его при себе
Kill your self

$0x6.8p-2$

↓

$110.1000 \times$

$\%a \leftrightarrow \%f$

можно .6 ← 6 цифр после точки

l ← размерность

Массивы

```
int q[10];  ← создание  (там ! мусор)

q[0] ... q[9]  ← обращение

q[10]  ← бог знает что будет
```

ровно что дают

```
int q[10] = {1, 2, 3}  ← инициализация  (первые 3: 1,2,3 ! остальные нули)

X q = {1,2,3}  ← не компилится

int q[];  ← 1 элемент

sizeof(q)/sizeof(q[0])  ← # элементов

int q[10] = {[2]=3, [1]=4}  <=> {0,4,3,0...0}
             idx   el

int q[2][3];
```

! двумерный массив – непрерывная область памяти

[0][0]	[0][1]	[0][2]
[1][0]	[1][1]	[1][2]

q[0]

[0][0], [0][1], [0][2], [1][0], [1][1], [1][2]

в памяти

нельзя
int q[x][3], x-должен быть константой вр. комп.

int w[0]; ✗
int main()
{
  int q[2][3]; ✓ создается на стеке => если q локал, то умрет
  static int w[x]; ✗
}

VLA

Статические и глобальные переменные всегда инициализируются

Указатели

int a=2, b;

int *p; p → указатель, p - указывает на int

p = &a; унарный амперсанд (взятие адреса)

&5 ✗ const; &(a+b) ✗ const; &a ✓

int q[2]

p = &q[1] ✓

float f;

p = &f; ✗ float

p = &a;

*p = 1; запись сквозь указатель
↓
a = 1

a = p ✗

const int c = 4;

int *p;

*p = 1 ← будут проблемы ∴ (изменит неизвестно что)

p = &c ← не скомпилируется тк const

const int *p;

p = &c; ✓

*p = 1; ✗

p = &a; ✓

const int *p = &a;

int *const p = &a;

p = &b; ✗

Но *p = 1; ✓

int *p, x; <=> int *p, x;

x - int ; p - указатель

int *p;

int **z = &p;

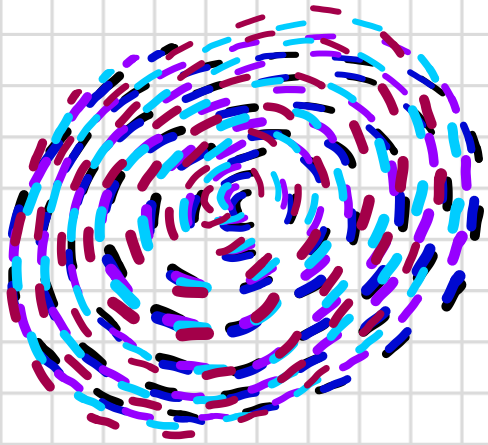
*z = &a; // p = &a

**z = 3; // a = 3

Void *V;

V = p или V = z или V = fp (float *fp;)

V = &c; ✗

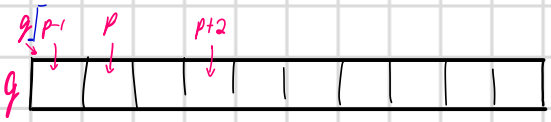


В C законно: p=v / z=v / fp=v

```
int q[2];
int *p;
*q=p // q[0]=1
```
указывает на первое

p=q; ✓ но q=p; ✗

```
p[0]=2;
p[3]=4;
p=&q[1];
p[0]=2; // q[1]=2
p[3]=4; // q[4]=4
p[-1]=2; // q[0]=2
```
p+2 (смещение на 2 эл. вперед)

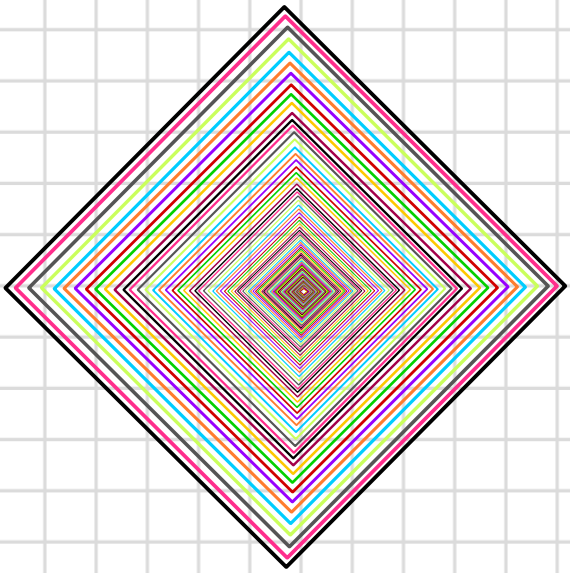


*(p+2) ≡ p[2] ✓

2+p ≡ 2[p] => 2[q] ✓

```
p=&*(q+1)
p=&q[1]; // p=q+1
p-q // 1 (расстояние между указателями)
q-p // -1
```

```
int *p;
p=malloc(sizeof(int)*100); // выделение памяти
```
(байты)



free(p); освобождение (далее Null)

Обязательно: if (!p) → error

void f(int a, int w[10]) — `w[10]` → `*w`, любое число

Нельзя передать массив, передаётся указатель

```
int main
{
    int q[10], b=4;
    f(b, q);
}

void f( int a, int w[10])
{
    w[3] = 1;   // изменит q[3]=1
    a = 2;      // не изменит b
}
```

чтобы изменить a → `void f( int *a int w[10])` и `f(&b, q);`

```
*a = 2;
```

```
int main()
{
    //∃w×h int
    int *p = malloc( sizeof(int) * w * h);   // можно *p
    p[y*w+x]   // x, y
    f(h, w, p);
}

void f( uint w, uint h, int *z)
{
    z[y·w+x] = 25;
}
```

тики-таки, съел все бигмаки