

Программирование на C++



Минцифры
России

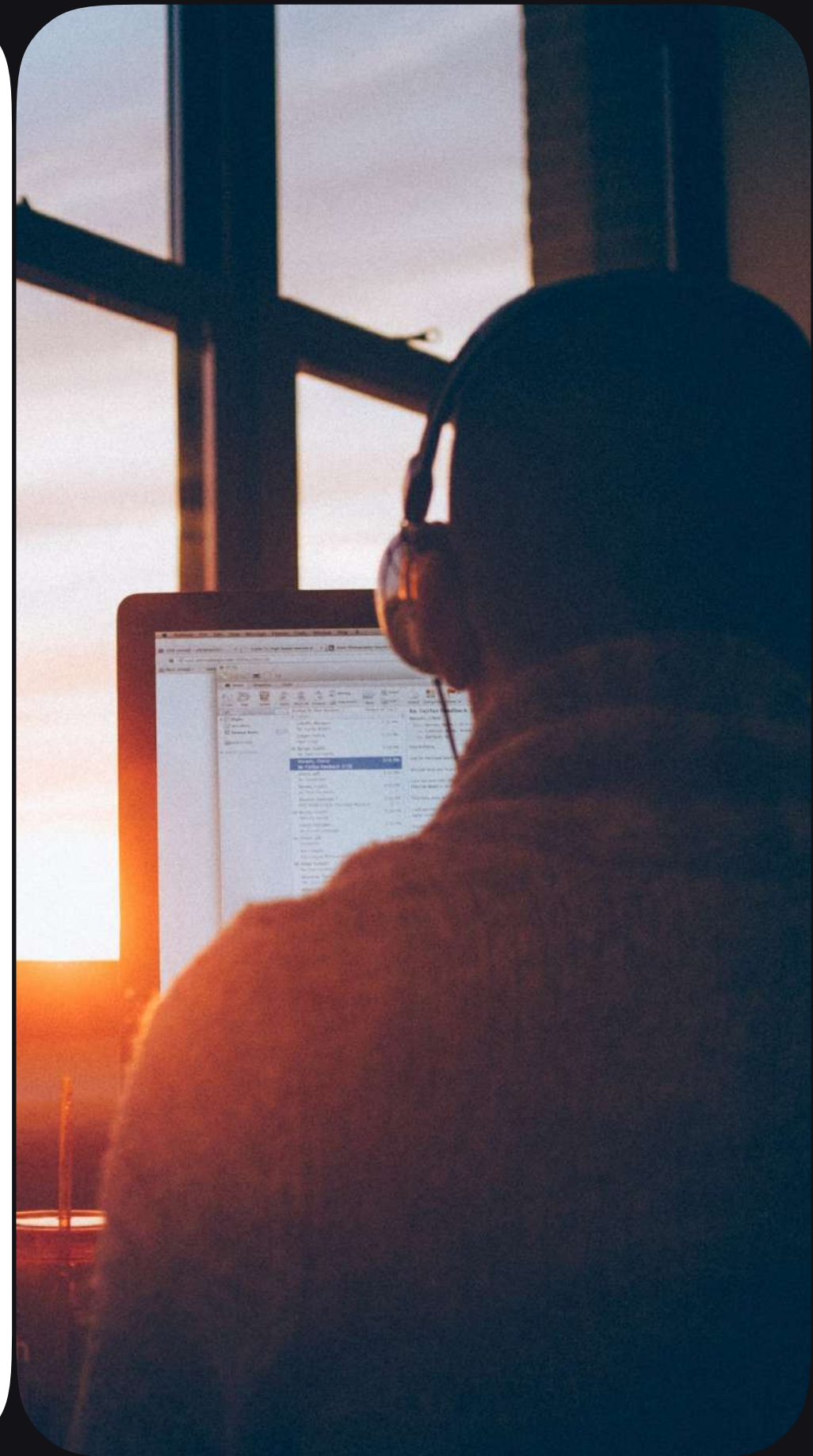
UCHi **DOMA**

20.35
УНИВЕРСИТЕТ

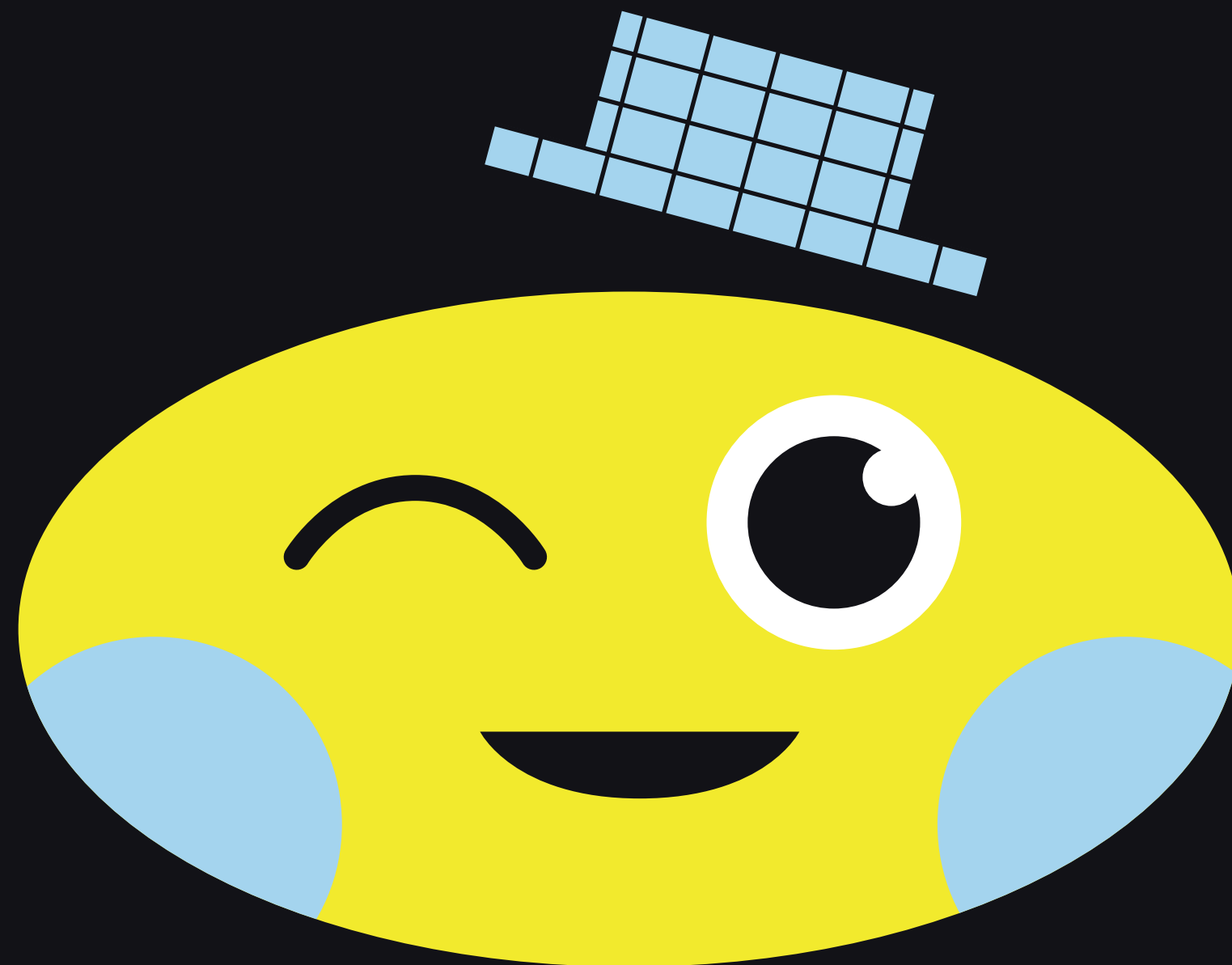


Модуль 1 Урок 12

Алгоритмы обработки массивов



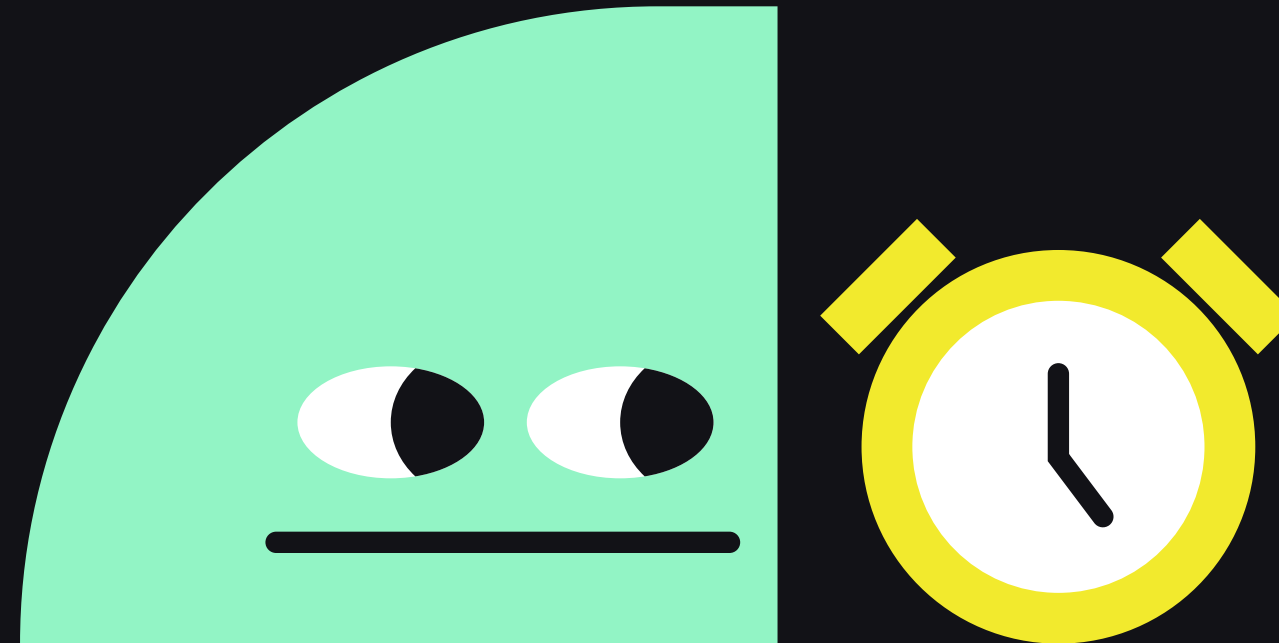
Привет!



проверка готовности



Видим и слышим друг друга без помех



Не опаздываем и не отвлекаемся



Сидим прямо



Улыбаемся, если всё ок

Как домашка?



Какие были трудности?



Какие остались вопросы?



Сколько заданий выполнено?



Разомнёмся



Что будет выведено на экран в результате работы программы?

```
1  #include <stdio.h>
2  int main()
3  {
4      int sum=0;
5      int m[3][3]={};
6      for (int i=0; i<3; i++)
7          for (int j=0; j<3; j++)
8              {
9                  m[i][j]=i*j;
10                 if (i==j) sum+=m[i][j]
11             }
12     printf("%d",sum);
13     return 0;
14 }
```

Разомнёмся



Что будет выведено на экран в результате работы программы?

```
1  #include <stdio.h>
2  int main()
3  {
4      int sum=0;
5      int m[3][3]={};
6      for (int i=0; i<3; i++)
7          for (int j=0; j<3; j++)
8              {
9                  m[i][j]=i*j;
10                 if (i==j) sum+=m[i][j]
11             }
12     printf("%d",sum);
13     return 0;
14 }
```

Результат работы программы:

5

0	0	0
0	1	2
0	2	4

Разомнёмся



Что будет выведено на экран в результате работы программы?

```
1  #include <stdio.h>
2  int main()
3  {
4      int m[3][3]={};
5      for (int i=0; i<3; i++)
6          for (int j=0; j<3; j++)
7              {
8                  m[i][j]=i*j;
9              }
10     printf("%d",m[1][1]);
11     return 0;
12 }
```


Разомнёмся



Что будет выведено на экран в результате работы программы?

```
1  #include <stdio.h>
2  int main()
3  {
4      int m[3][3]={};
5      for (int i=0; i<3; i++)
6          for (int j=0; j<3; j++)
7              {
8                  m[i][j]=i*j;
9              }
10     printf("%d",m[1][1]);
11     return 0;
12 }
```

Результат работы программы:

1

	0	1	2
0	0	0	0
1	0	1	2
2	0	2	4

Цели урока



изучить алгоритмы
обработки массивов



отработать на практике
составление алгоритмов
с хранением данных в массиве
на Си



Поиск максимального элемента

```
1  #include <stdio.h>
2  int main()
3  {
4      int m[5] = {1, -1, 0, 4, 2};
5      int max = m[0];
6      int i;
7      for(i=0; i<5; ++i)
8      {
9          if(m[i]>max)
10         {
11             max=m[i];
12         }
13     }
14     printf("Max=%d\n", max);
15     return 0;
16 }
```

Результат работы программы:

Max=4

Поиск минимального элемента

```
1  #include <stdio.h>
2  int main()
3  {
4      int m[5] = {1, -1, 0, 4, 2};
5      int min = m[0];
6      int i;
7      for(i=0; i<5; ++i)
8      {
9          if(m[i]<min)
10         {
11             min=m[i];
12         }
13     }
14     printf("Min=%d\n", min);
15     return 0;
16 }
```

Результат работы программы:

Min=-1

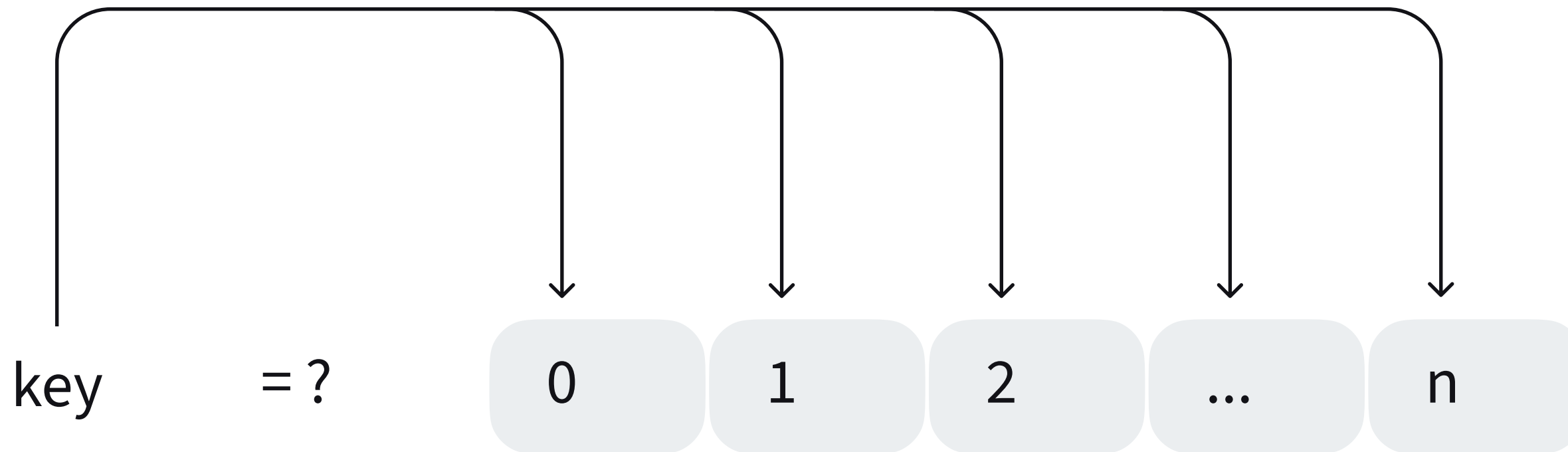
Линейный поиск



Алгоритм перебирает все элементы в массиве, сравнивая их с заданным ключом.



Из-за полного перебора отличается низкой скоростью поиска.



Линейный поиск



Инициировать массив значениями 1, -1, 0, 4, 2. Запросить у пользователя значение ключа. Осуществить линейный поиск ключа в массиве, если искомое значение найдено — вывести индекс.

```
1  #include <stdio.h>
2  int main()
3  {
4      int m[5] = {1, -1, 0, 4, 2};
5      int key;
6      scanf("%d", &key);
7      for(int i=0; i<5; ++i)
8          if(m[i]==key)
9              printf("%d\n", i);
10     return 0;
11 }
```

Линейный поиск

Бинарный поиск



Бинарный поиск работает только в **отсортированном массиве**.



При бинарном поиске искомый ключ сравнивается с ключом среднего элемента в массиве. Если они равны, то поиск успешен. В противном случае поиск осуществляется аналогично в левой или правой частях массива.



Бинарный поиск также называют поиском методом **деления отрезка пополам** или **дихотомии**.

Бинарный поиск

На каждом шаге осуществляется поиск середины отрезка по формуле:

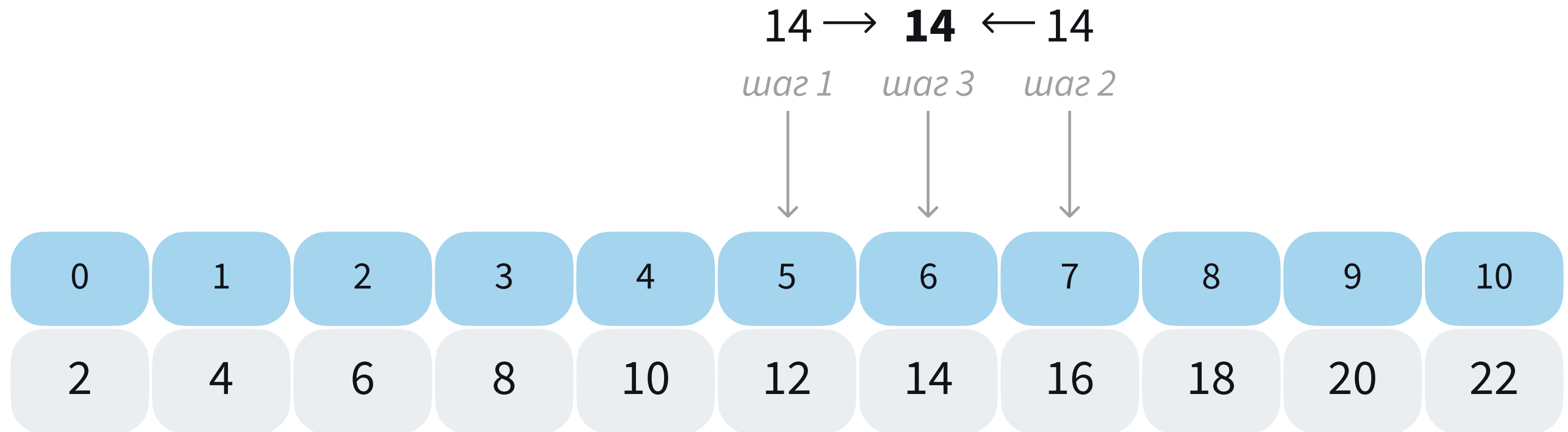
$$\text{mid} = (\text{left} + \text{right}) / 2$$

Если искомый элемент равен элементу с индексом **mid**, поиск завершается.

В случае если искомый элемент меньше элемента с индексом **mid**, на место **mid** перемещается правая граница рассматриваемого отрезка, в противном случае — левая граница.

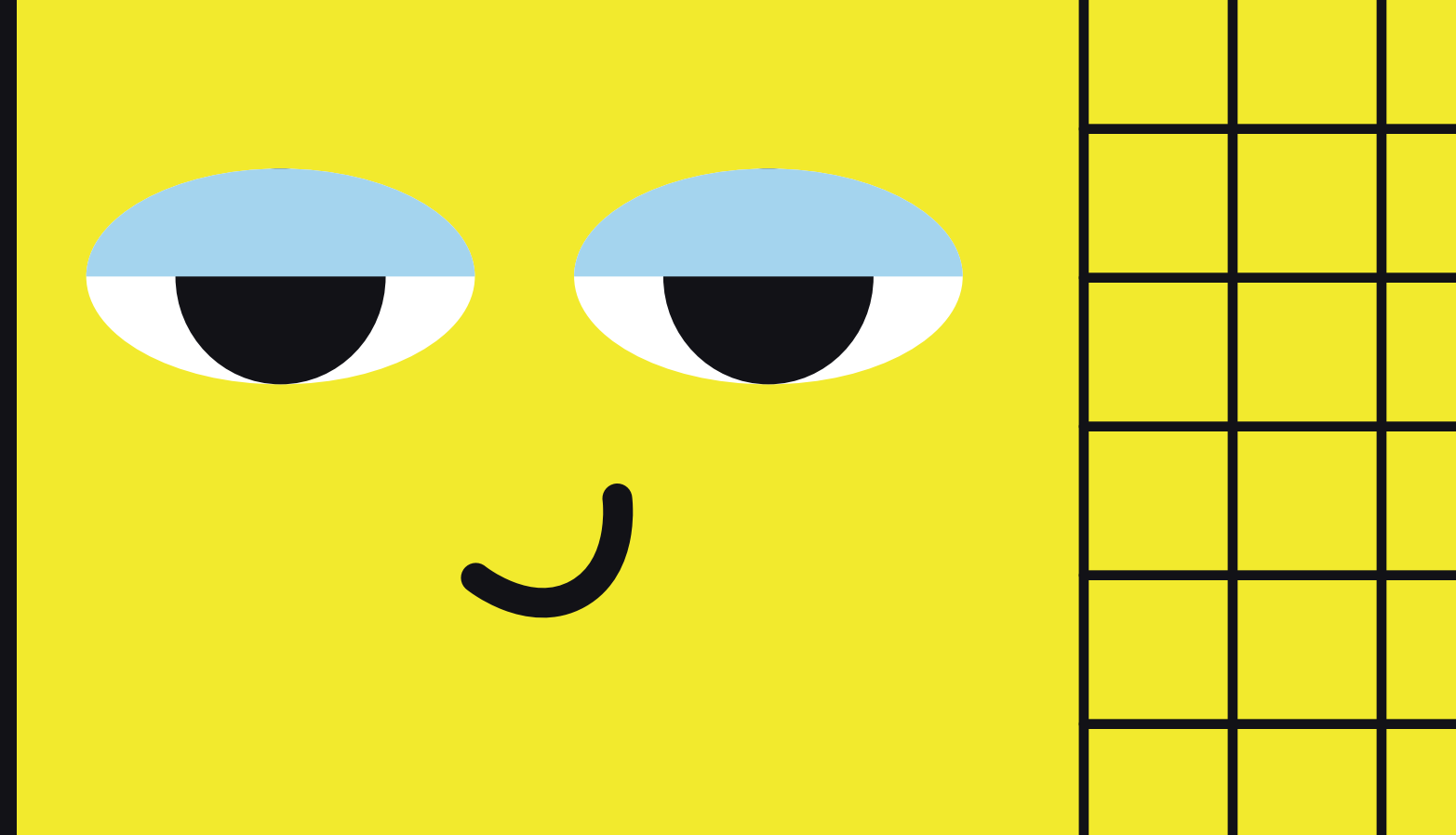
Бинарный поиск

Будем искать 14



Бинарный поиск

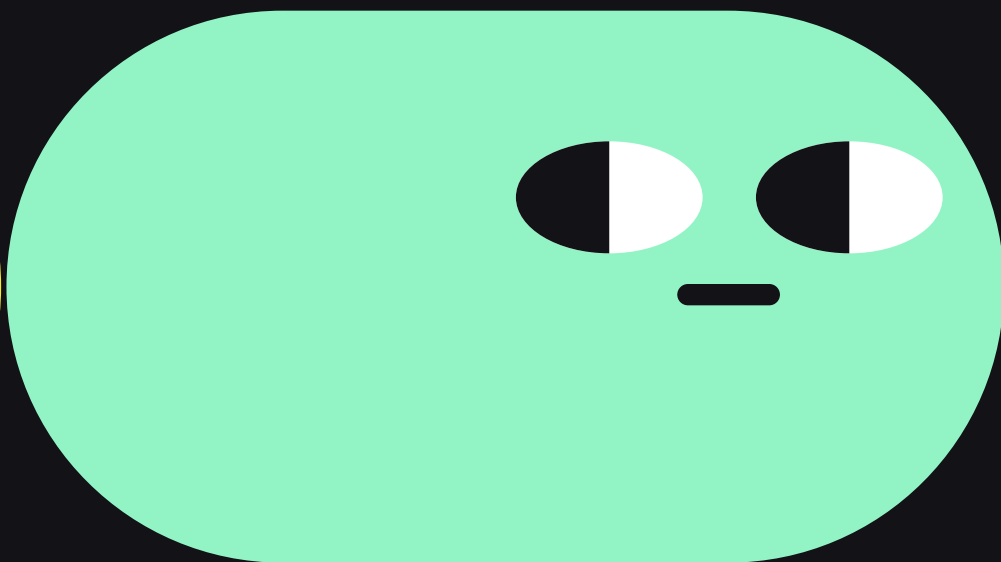
```
1  #include <stdio.h>
2  int main()
3  {
4      int k[11]={2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22};
5      int key, index;
6      printf("Введите key: "); // вводим искомое ключевое поле
7      scanf("%d", &key);
8      int left = 0; // задаем левую и правую границы поиска
9      int right = 10;
10     int search = -1; // найденный индекс элемента равен -1 (элемент не найден)
11     while (left <= right) // пока левая граница не "перескочит" правую
12     {
13         int mid = (left + right) / 2; // ищем середину отрезка
14         if (key == k[mid]) { // если ключевое поле равно искомому
15             search = mid; // мы нашли требуемый элемент,
16             index=mid;
17             break; // выходим из цикла
18         }
19         if (key < k[mid]) // если искомое ключевое поле меньше найденной середины
20             right = mid - 1; // смещаем правую границу, продолжим поиск в левой части
21         else // иначе
22             left = mid + 1; // смещаем левую границу, продолжим поиск в правой части
23     }
24     if (search == -1) // если индекс элемента по-прежнему -1, элемент не найден
25         printf("Элемент не найден!\n");
26     else // иначе выводим элемент, его ключ и значение
27         printf("Номер искомого элемента: %d", index);
28     return 0;
29 }
```



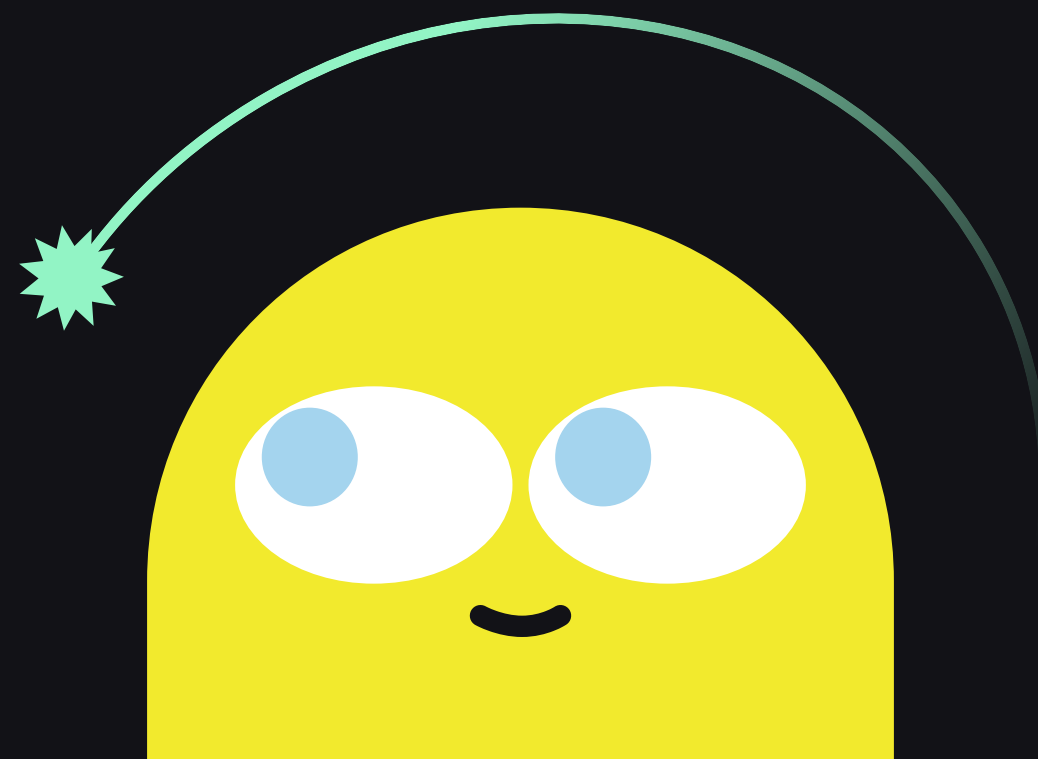
Практика

перерыв

физкультминутка



Смотрим вверх–вниз, вправо–влево



Вращаем по кругу туда–обратно



Крепко зажимаемся



Быстро моргаем

Сортировки

Сортировка — это упорядочение элементов массива по возрастанию либо по убыванию.

Прямые методы

✦ Сортировка прямыми включениями

✦ Сортировка прямым выбором

✦ Сортировка прямым обменом
(метод «пузырька»)

✦ Сортировка с помощью дерева

Улучшенные методы

✦ Пирамидальная сортировка

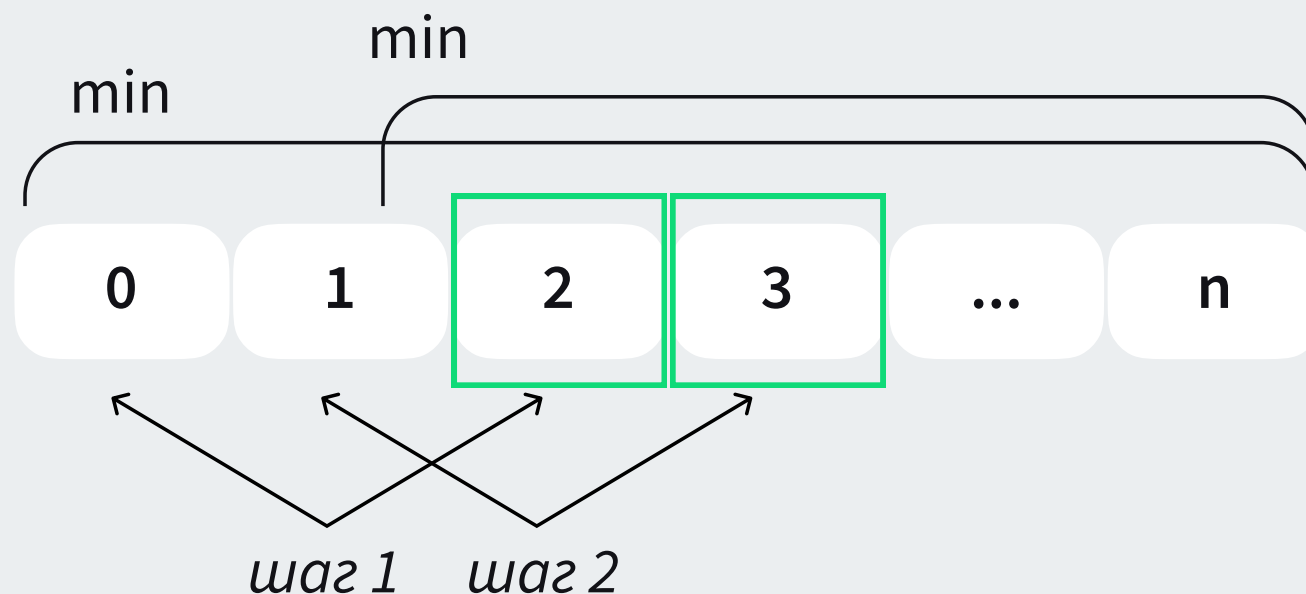
✦ Быстрая сортировка (Ч.Хоар)

✦ Шейкер-сортировка

Сортировка прямым выбором

Сортируем по возрастанию значений элементов массива:

- ✦ Выбирается элемент с наименьшим значением.
- ✦ Он меняется местами с первым элементом a_0 .
- ✦ Затем эти операции повторяются с оставшимися $n-1$ элементами, $n-2$ элементами и так далее до тех пор, пока не останется один, самый большой элемент.



Сортировка прямым выбором

```
1  #include <stdio.h>
2  int main()
3  {
4      int a[10]={5, 1, 3, 8, 4, 9, 2, 6, 0, 7};
5      int min, temp; // для поиска минимального элемента и для обмена
6      for (int i=0; i<10-1; i++)
7      {
8          min = i; // запоминаем индекс текущего элемента
9          // ищем минимальный элемент чтобы поместить на место i-ого
10         for (int j=i+1; j<10; j++) // для остальных элементов после i-ого
11         {
12             if (a[j] < a[min]) // если элемент меньше минимального,
13                 min = j; // запоминаем его индекс в min
14         }
15         temp = a[i]; // меняем местами i-ый и минимальный элементы
16         a[i] = a[min];
17         a[min] = temp;
18     }
19     for (int i = 0; i<10; i++)
20         printf("%d", a[i]);
21     return 0;
22 }
```

Результат работы программы:

0 1 2 3 4 5 6 7 8 9

Сортировка прямым обменом (метод «пузырька»)



Алгоритм основан на принципе **сравнения и обмена** пары соседних элементов до тех пор, пока не будут отсортированы все элементы.

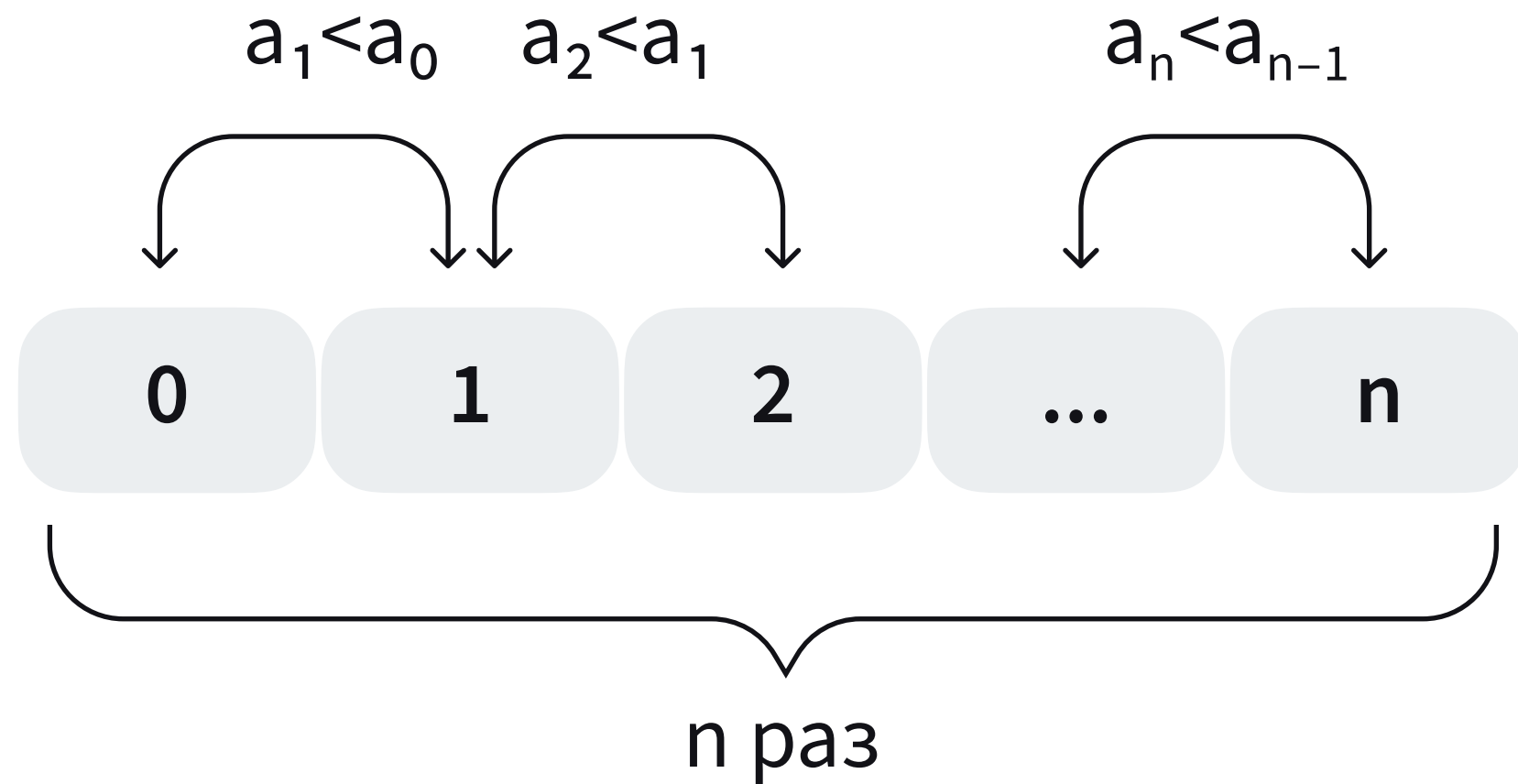


Совершаются проходы по массиву, сдвигая каждый раз наименьший элемент оставшейся последовательности к началу массива.



Таким образом наименьшие элементы **«всплывают»** как пузырьки воздуха в жидкости, постепенно двигаясь к началу массива.

Сортировка прямым обменом (метод «пузырька»)

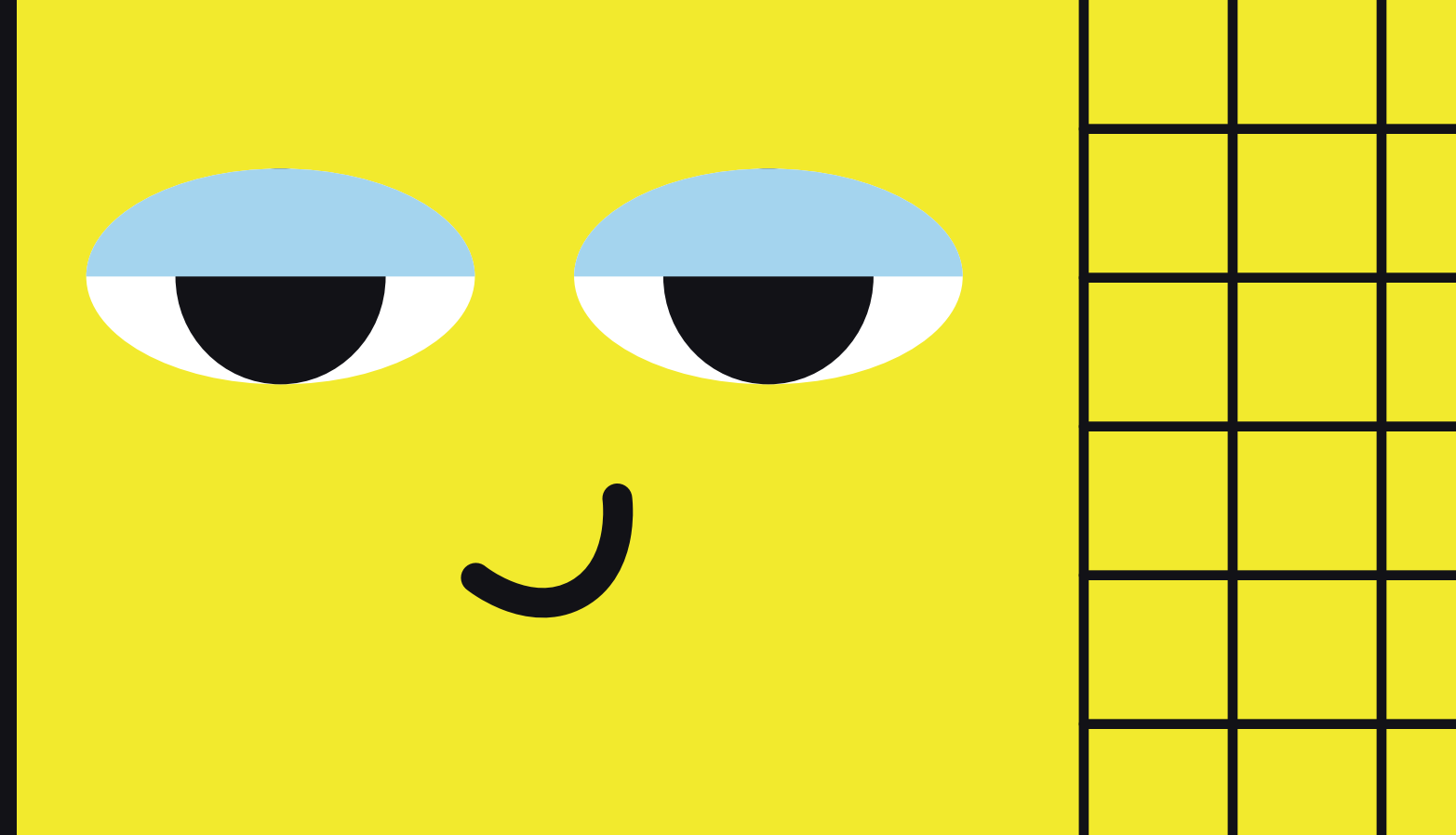


Сортировка прямым обменом (метод «пузырька»)

```
1  #include <stdio.h>
2  int main()
3  {
4      int a[10]={5, 1, 3, 8, 4, 9, 2, 6, 0, 7};
5      // Для всех элементов
6      for (int i=0; i<10-1; i++)
7      {
8          for (int j=(10-1); j>i; j--) // для всех элементов после i-ого
9          {
10             if (a[j-1] > a[j]) // если текущий элемент меньше предыдущего
11             {
12                 int temp = a[j-1]; // меняем их местами
13                 a[j-1] = a[j];
14                 a[j] = temp;
15             }
16         }
17     }
18     // Выводим отсортированные элементы массива
19     for (int i=0; i<10; i++)
20         printf("%d", a[i]);
21     return 0;
22 }
```

Результат работы
программы:

0 1 2 3 4 5 6 7 8 9



Практика

1

Закрепление

Что делает программа?

```
1  #include <stdio.h>
2  int main()
3  {
4      int m[5] = {1, -1, 0, 4, 2};
5      int max = m[0];
6      int i;
7      for(i=0; i<5; ++i)
8      {
9          if(m[i]<max)
10         {
11             max=m[i];
12         }
13     }
14     printf("%d\n", max);
15     return 0;
16 }
```

1

Закрепление

Что делает программа?

```
1  #include <stdio.h>
2  int main()
3  {
4      int m[5] = {1, -1, 0, 4, 2};
5      int max = m[0];
6      int i;
7      for(i=0; i<5; ++i)
8      {
9          if(m[i]<max)
10         {
11             max=m[i];
12         }
13     }
14     printf("%d\n", max);
15     return 0;
16 }
```

Результат работы программы:

-1



Программа находит
наименьший
элемент массива

Подведём итоги



изучили алгоритмы
обработки массивов



отработали на практике
составление алгоритмов
с хранением данных
в массиве на Си



Оцени сложность урока

1

было совсем
просто

2

было достаточно
просто, но ты
узнал(а) что-то
новое

3

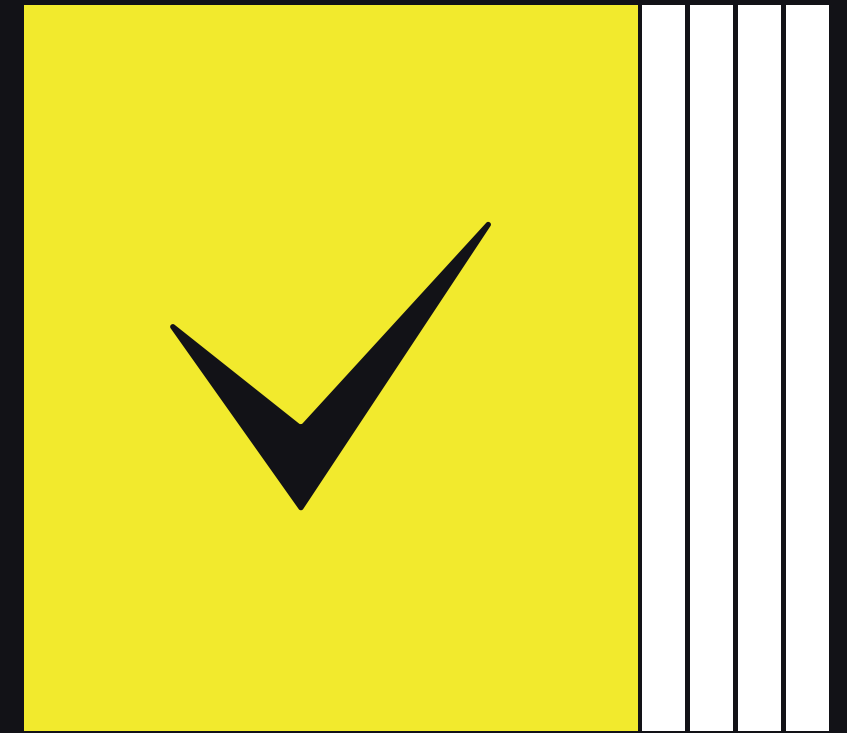
было не очень
просто, но
интересно

4

было сложно,
не знал(а) ничего
из материала

5

было слишком
сложно, многое
осталось
непонятным



**Мы завершили
изучение
первого модуля!**

В первом модуле мы изучили:



История С. Основные понятия



Операторы ввода, вывода



Операторы ветвления в неполной форме



Оператор ветвления в полной форме



Вложенный оператор ветвления



Оператор множественного выбора



Создали проект «Текстовый квест»



Цикл с преусловием и постусловием



Цикл со счётчиком



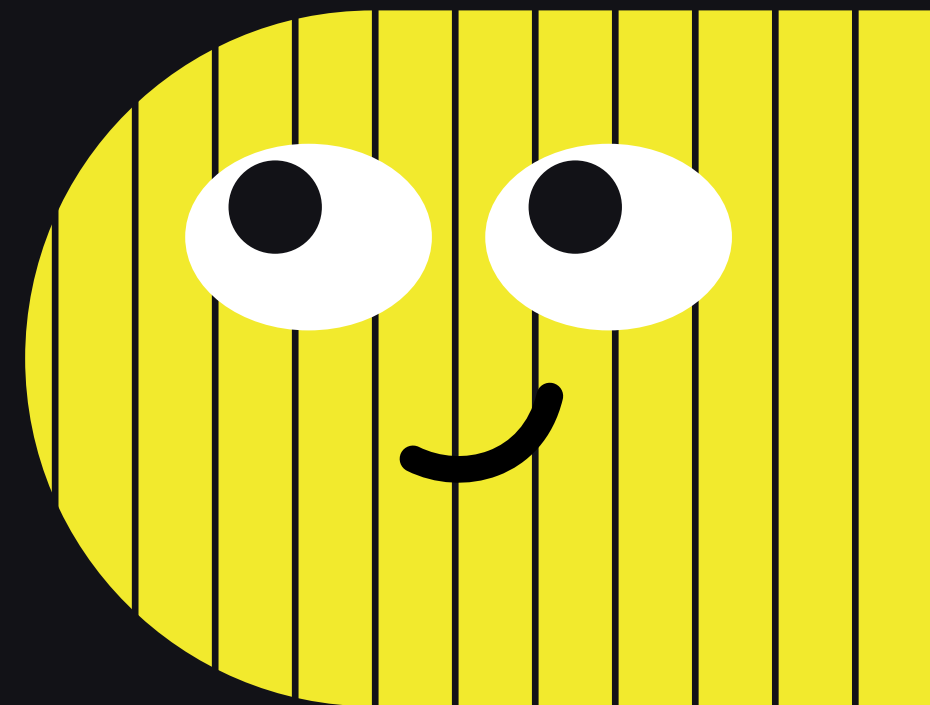
Массивы



Двумерные массивы



Алгоритмы обработки массивов



Домашнее задание – практика



До встречи!