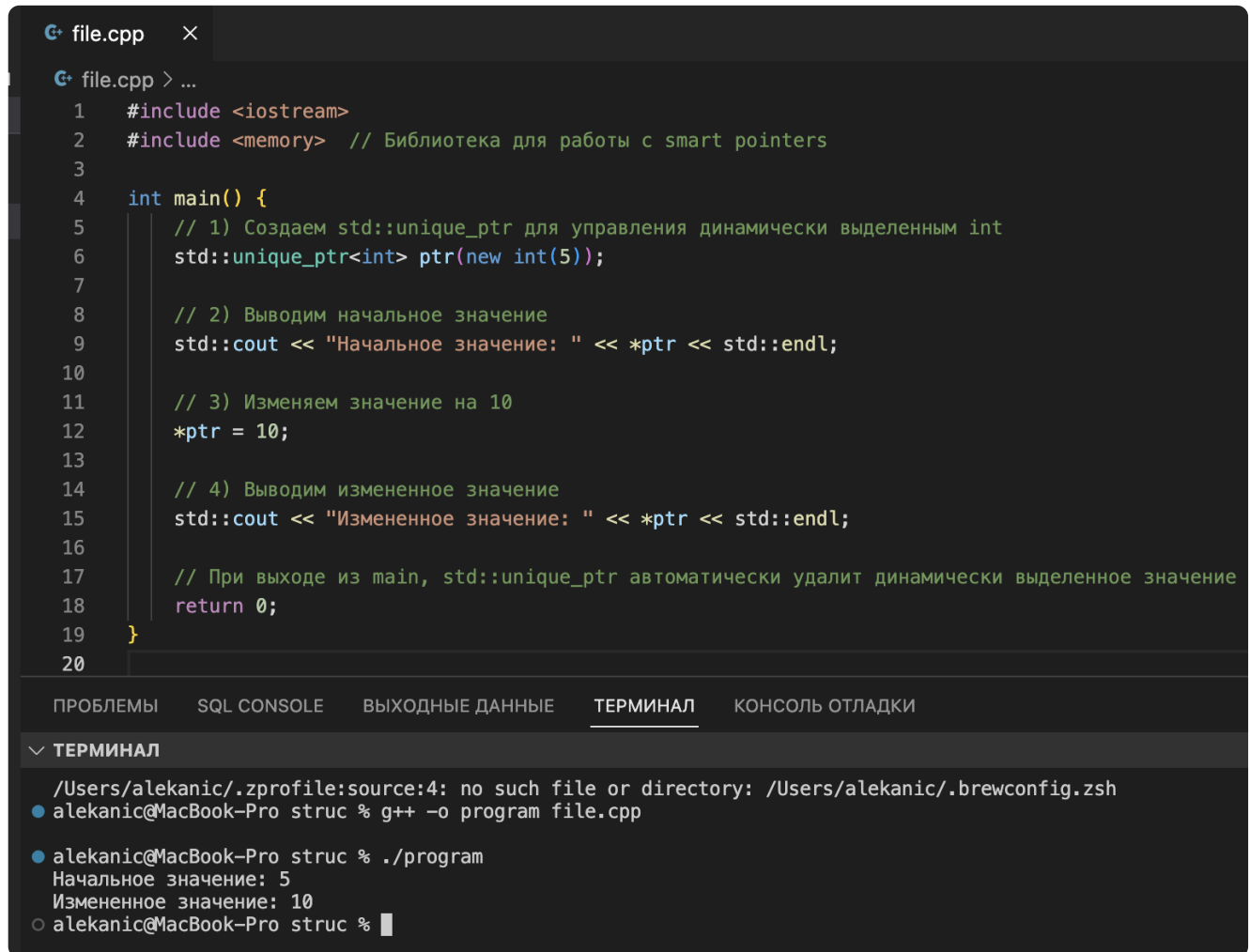


Структуры и алгоритмы компьютерной обработки данных

Задание 1



```
file.cpp  ×
file.cpp > ...
1  #include <iostream>
2  #include <memory> // Библиотека для работы с smart pointers
3
4  int main() {
5      // 1) Создаем std::unique_ptr для управления динамически выделенным int
6      std::unique_ptr<int> ptr(new int(5));
7
8      // 2) Выводим начальное значение
9      std::cout << "Начальное значение: " << *ptr << std::endl;
10
11     // 3) Изменяем значение на 10
12     *ptr = 10;
13
14     // 4) Выводим измененное значение
15     std::cout << "Измененное значение: " << *ptr << std::endl;
16
17     // При выходе из main, std::unique_ptr автоматически удалит динамически выделенное значение
18     return 0;
19 }
20
```

ПРОБЛЕМЫ SQL CONSOLE ВЫХОДНЫЕ ДАННЫЕ ТЕРМИНАЛ КОНСОЛЬ ОТЛАДКИ

▼ ТЕРМИНАЛ

```
/Users/alekanic/.zprofile:source:4: no such file or directory: /Users/alekanic/.brewconfig.zsh
● alekanic@MacBook-Pro struc % g++ -o program file.cpp

● alekanic@MacBook-Pro struc % ./program
Начальное значение: 5
Измененное значение: 10
○ alekanic@MacBook-Pro struc %
```

Задание 2

file.cpp

file.cpp > main()

```
1  #include <iostream>
2  #include <list>
3
4  int main() {
5      // 1) Подключение необходимых библиотек выполнено (см. выше)
6
7      // 2) Создаем пустой список и переменную для хранения вводимых значений
8      std::list<int> myList;
9      int value;
10
11     // 3) Вводим 5 элементов и добавляем их в список
12     std::cout << "Введите 5 элементов списка: " << std::endl;
13     for (int i = 0; i < 5; ++i) {
14         std::cin >> value;
15         myList.push_back(value);
16     }
17
18     // 4) Создаем переменную для хранения значения, которое нужно удалить
19     int valueToRemove;
20     std::cout << "Введите значение для удаления: ";
21     std::cin >> valueToRemove;
22
23     // 5) Удаляем указанные элементы и выводим обновленный список
24     myList.remove(valueToRemove);
25
26     std::cout << "Обновленный список: ";
27     for (int elem : myList) {
28         std::cout << elem << " ";
29     }
30     std::cout << std::endl;
31
32     return 0;
33 }
34
```

ПРОБЛЕМЫ

SQL CONSOLE

ВЫХОДНЫЕ ДАННЫЕ

ТЕРМИНАЛ

КОНСОЛЬ ОТЛАДКИ

▼ ТЕРМИНАЛ

```
● alekanic@MacBook-Pro struc % ./program
Введите 5 элементов списка:
1 2 3 4 5
Введите значение для удаления: 5
Обновленный список: 1 2 3 4
○ alekanic@MacBook-Pro struc %
```

Задание 3

```

file.cpp > isPalindrome(const std::string &)
1  #include <iostream>
2  #include <stack>
3  #include <string>
4
5  // Функция для проверки, является ли строка палиндромом
6  bool isPalindrome(const std::string& str) {
7      std::stack<char> s;
8      int n = str.length();
9
10     // Добавляем первую половину строки в стек
11     for (int i = 0; i < n / 2; ++i) {
12         s.push(str[i]);
13     }
14
15     // Определяем начало второй половины
16     int start = (n + 1) / 2;
17
18     // Сравниваем вторую половину с элементами стека
19     for (int i = start; i < n; ++i) {
20         if (s.top() != str[i]) {
21             return false; // Символы не равны, значит не палиндром
22         }
23         s.pop(); // Удаляем верхний элемент стека
24     }
25
26     return true; // Все символы совпали, это палиндром
27 }
28
29 int main() {
30     // Создаем переменную для хранения строки
31     std::string input;
32
33     // Ввод строки от пользователя
34     std::cout << "Введите строку для проверки на палиндром: ";
35     std::getline(std::cin, input);
36
37     // Проверка строки на палиндром и вывод результата
38     if (isPalindrome(input)) {
39         std::cout << "Строка является палиндромом." << std::endl;
40     } else {
41         std::cout << "Строка не является палиндромом." << std::endl;
42     }
43
44     return 0;
45 }
46

```

ПРОБЛЕМЫ SQL CONSOLE ВЫХОДНЫЕ ДАННЫЕ ТЕРМИНАЛ КОНСОЛЬ ОТЛАДКИ

▼ ТЕРМИНАЛ

- alekanic@MacBook-Pro struc % ./program
Введите строку для проверки на палиндром: level
Строка является палиндромом.
- ⊗ alekanic@MacBook-Pro struc % строка
zsh: command not found: строка
- alekanic@MacBook-Pro struc % ./program
Введите строку для проверки на палиндром: строка
Строка не является палиндромом.

Задание 4

```
file.cpp  X
file.cpp > ...
1  #include <iostream>
2  #include <thread>
3  #include <mutex>
4
5  int counter = 1; // Глобальная переменная-счётчик
6  std::mutex mtx; // Мьютекс для синхронизации
7
8  void printEven() {
9      while (counter <= 10) {
10         std::lock_guard<std::mutex> lock(mtx); // Автоматическая блокировка мьютекса
11         if (counter % 2 == 0) {
12             std::cout << "Чётное число: " << counter << std::endl;
13             ++counter;
14         }
15     }
16 }
17
18 void printOdd() {
19     while (counter <= 10) {
20         std::lock_guard<std::mutex> lock(mtx); // Автоматическая блокировка мьютекса
21         if (counter % 2 != 0) {
22             std::cout << "Нечётное число: " << counter << std::endl;
23             ++counter;
24         }
25     }
26 }
27
28 int main() {
29     // Создаем два потока
30     std::thread t1(printOdd);
31     std::thread t2(printEven);
32
33     // Ожидаем завершения потоков
34     t1.join();
35     t2.join();
36
37     return 0;
38 }
39
```

ПРОБЛЕМЫ SQL CONSOLE ВЫХОДНЫЕ ДАННЫЕ ТЕРМИНАЛ КОНСОЛЬ ОТЛАДКИ

▼ ТЕРМИНАЛ

```
● alekanic@MacBook-Pro struc % g++ -o program file.cpp
● alekanic@MacBook-Pro struc % ./program
Нечётное число: 1
Чётное число: 2
Нечётное число: 3
Чётное число: 4
Нечётное число: 5
Чётное число: 6
Нечётное число: 7
Чётное число: 8
Нечётное число: 9
Чётное число: 10
Нечётное число: 11
```

Задание 5

```
1  #include <iostream>
2
3  // Структура узла односвязного списка
4  struct Node {
5      int data;
6      Node* next;
7      Node(int value) : data(value), next(nullptr) {}
8  };
9
10 // Класс для односвязного списка
11 class LinkedList {
12 public:
13     LinkedList() : head(nullptr) {}
14
15     // Функция для добавления элемента в начало списка
16     void push_front(int value) {
17         Node* newNode = new Node(value);
18         newNode->next = head;
19         head = newNode;
20     }
21
22     // Функция для сортировки списка методом вставками
23     void insertionSort() {
24         if (!head || !head->next) {
25             return;
26         }
27
28         Node* sorted = nullptr; // Сортированная часть списка
29         Node* current = head;
30
31         while (current != nullptr) {
32             Node* next = current->next;
33             sortedInsert(&sorted, current);
34             current = next;
35         }
36         head = sorted;
37     }
38
39     // Функция для вставки узла в сортированную часть списка
40     void sortedInsert(Node** sorted, Node* newNode) {
41         if (*sorted == nullptr || (*sorted)->data >= newNode->data) {
42             newNode->next = *sorted;
43             *sorted = newNode;
44         } else {
45             Node* current = *sorted;
46             while (current->next != nullptr && current->next->data < newNode->data) {
47                 current = current->next;
48             }
49             newNode->next = current->next;
50             current->next = newNode;
51         }
52     }
53 }
```

```

52     }
53
54     // Функция для печати списка
55     void printList() const {
56         Node* current = head;
57         while (current != nullptr) {
58             std::cout << current->data << " ";
59             current = current->next;
60         }
61         std::cout << std::endl;
62     }
63
64     // Деструктор для очистки списка
65     ~LinkedList() {
66         while (head != nullptr) {
67             Node* temp = head;
68             head = head->next;
69             delete temp;
70         }
71     }
72
73 private:
74     Node* head;
75 };
76
77 int main() {
78     LinkedList list;
79     list.push_front(4);
80     list.push_front(3);
81     list.push_front(2);
82     list.push_front(10);
83     list.push_front(12);
84     list.push_front(1);
85     list.push_front(5);
86
87     std::cout << "Несортированный список: ";
88     list.printList();
89
90     list.insertionSort();
91
92     std::cout << "Сортированный список: ";
93     list.printList();
94
95     return 0;
96 }
97
98

```

ПРОБЛЕМЫ

SQL CONSOLE

ВЫХОДНЫЕ ДАННЫЕ

ТЕРМИНАЛ

КОНСОЛЬ ОТЛАДКИ

▼ ТЕРМИНАЛ

- alekanic@MacBook-Pro struc % g++ -o program file.cpp
- alekanic@MacBook-Pro struc % ./program

Несортированный список: 5 1 12 10 2 3 4
 Сортированный список: 1 2 3 4 5 10 12
- alekanic@MacBook-Pro struc %

