

Министерство образования Российской Федерации
Пензенский государственный университет
Кафедра «Вычислительная техника»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе
по курсу «Программирование»
на тему «База данных»

Выполнил:

студент группы 18ВВ2
Лехов К. А.

Принял:

к.т.н., доцент Федюнин Р.Н.

Пенза, 2019

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Факультет Вычислительной техники

Кафедра "Вычислительная техника"

"УТВЕРЖДАЮ"

Зав. кафедрой ВТ

Митрохин М.А.



« ____ » _____ 20__

ЗАДАНИЕ

на курсовое проектирование по курсу

Программирование

Студенту Лехову Кириллу Александровичу Группа 18ВВ2

Тема проекта: Разработка программы сложной структуры методом нисходящего программирования

Исходные данные (технические требования) на проектирование

Наименование программы: База данных *Готовые операторы*

База данных включает следующие поля:

Задание: Программа должна выполнять следующие функции:

- Создание новой базы данных. Добавление записей
- Удаление записей. Редактирование записей
- Поиск и сортировка записей. Фильтрация записей по определенному критерию.
- Сохранение текущей базы данных в файл. Загрузка сохраненных баз данных из файла.

Обязательные требования к программе:

1. Многомодульность.
2. Использование сложных типов данных – списков, структур, файлов.
3. Режим работы видеосистемы – текстовый/графический.
4. Устройство ввода информации – клавиатура, мышь.
5. Пользовательский интерфейс должен быть построен на основе меню и панели инструментов.
6. Наличие заставки.
7. Операционная система – MS Windows
8. Система программирования – Си/Си++ на базе Среды Microsoft Visual Studio 2015.
9. Язык программирования – Си и Ассемблер

Расчетная часть

Разработка программы

Графическая часть

Схема данных, схема ресурсов системы, схема работы системы, иерархическая структура программы, схема взаимодействия программы, схема программы.

Экспериментальная часть

Отладка программы

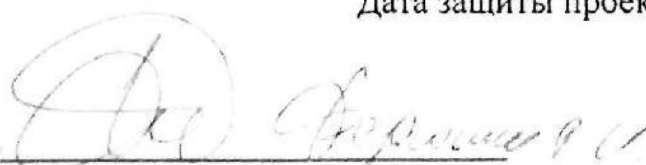
Срок выполнения проекта по разделам

1. В соответствии с графиком выполнения курсового проекта
2. _____
3. _____
4. _____
5. _____

Дата выдачи задания " 28 " 01 2019 г.

Дата защиты проекта " ____ " _____

Руководитель



Задание получил " 28 " 01 2019 г.

Студент Иванов И. А.

Содержание

Введение	5
Постановка задачи	7
Выбор решения	8
Описание разработки программы	9
Отладка и тестирование	10
Описание программы	10
Руководство пользователя	26
Приложение А Листинги программы	44
Приложение В Результаты работы программы	78
Приложение С Результат работы программы	80

Введение

На сегодняшний день информация – ценный источник знаний, систематизации которой, позволяет оптимизировать работу как человека, так и общества в целом. Компьютерные технологии во многом упрощают нашу жизнь в самых разнообразных сферах нашей деятельности. И именно компьютерные технологии позволяют грамотно взаимодействовать с информацией, обрабатывать её и хранить на компактных носителях.

База данных представляет собой упорядоченную, структурированную совокупность данных, подобранных таким образом, что эти данные могут легко и быстро обрабатываться на ЭВМ.

История баз данных в узком смысле рассматривает базы данных в традиционном (современном) понимании. Эта история начинается с 1955 года, когда появилось программируемое оборудование обработки записей. Программное обеспечение этого времени поддерживало модель обработки записей на основе файлов. Для хранения данных использовались перфокарты.

Оперативные сетевые базы данных появились в середине 1960-х. Операции над оперативными базами данных обрабатывались в интерактивном режиме с помощью терминалов. Простые индексно-последовательные организации записей быстро развились к более мощной модели записей, ориентированной на наборы.

В наше время базы данных являются неотъемлемой частью любой организации, любого предприятия. Динамические сайты используют базу данных чтобы хранить данные пользователя, как и стриминговые сервисы с подписочной моделью. Банки хранят в огромных базах данных данные пользователей.

Данная программа является информационно-поисковой системой и позволяет работать с простейшей базой данных сотовых операторов, позволяющей любому человек подобрать выгодный тарифный план под свои нужды, отслеживать историю тарифных планов сотовых операторов и наблюдать историю их изменения.

Постановка задачи

Необходимо разработать программу для работы с базой данных сотовых операторов. Программа должна иметь интуитивно понятный интерфейс для пользователя. Необходимо определить названия пунктов меню, построить алгоритмы работы с массивами структур для фильтрации и сортировки записей по определенным критериям, изучить функции для работы с файлами, чтобы сохранять базу данных в файл и загружать из файла.

Многомодульность программы. Программа должна быть поделена на логические модули. Это упростит поиск ошибок при отладке и тестировании программы, а также позволит легко расширять функционал программы.

Использование сложных типов данных – массивов, структур, файлов. Это необходимо для более простой и интуитивной обработки данных в коде программы.

Режим работы видеосистемы – текстовый/графический. Для начала необходимо определиться с типом интерфейса и с элементами управления, затем необходимо изучить способы их реализации.

Устройство ввода-вывода – клавиатура и мышь. Необходимо различать и идентифицировать действия, произведенные с их помощью, это облегчит использование программы.

Интерфейс должен быть построен на основе меню. Это необходимо для создания интуитивно понятного интерфейса для пользователя.

Заставка необходима для того, чтобы пользователь, запустивший программу, смог получить достаточную информацию о ней.

Выбор решения

При запуске программы открывается приветственный экран. Далее появляется меню, состоящее из семи пунктов:

- 1) Добавить запись. Пользователь вводит данные, необходимые для нового элемента базы данных в порядке, указанном в программе. Программа ищет записи с ID = -1. Если такая имеется, то данные из её полей заменяются на данные пользователя и ставит ID > 0. Если записи с ID = -1 не найдены, то программа добавляет новую запись в базу данных.
- 2) Вывести записи. Пользователь выбирает один пункт из нового меню:
 - 2.1 Показать все записи. На экране отобразится вся база данных.
 - 2.2 Поиск по фильтру. Пользователь выбирает один пункт из нового меню:
 - 2.2.1 Поиск по ID. Пользователь вводит ID записи. Если элемент с данным ID существует, то программа выводит его, иначе выдаётся ошибка.
 - 2.2.2 Поиск по оператору. Пользователь вводит полное наименование оператора или его часть. Программа ищет записи с подобными данными и отображает в виде таблицы на дисплее.
 - 2.2.3 Поиск по тарифу. Аналогично предыдущему пункту, только вместо наименования оператора выступает наименование тарифа.
 - 2.2.4 Поиск по стоимости. Пользователь вводит с клавиатуры стоимость услуги, далее символ, необходимый для отбора записей «>» или «<». Происходит подбор записей в соответствии с условием и их вывод на экране.
 - 2.2.5 Поиск по дате. Пользователь вводит дату с клавиатуры. Программа ищет записи с данной датой и выводит все элементы базы данных на экран.
 - 2.2.6 Поиск по доступности. Пользователь вводит с клавиатуры символ «Y», «y», «D», «d» - доступный тариф или «N», «n», «H», «h» - архивный тариф. Программа находит все записи, соответствующие критерию доступности, и выводит их на экран.
 - 2.3 Отсортировать записи. Пользователь выбирает один пункт из нового меню:
 - 2.3.1 Сортировать в обратном порядке по ID. Программа выводит все записи базы данных в обратном порядке (от последней к первой) на экран.
 - 2.3.2 Сортировать по цене. Программа выводит на экран все записи базы данных сортируя их по полю «Цена».
 - 2.3.3 Сортировать по цене в обратном порядке. Программа выводит на экран все записи базы данных сортируя их по полю «Цена» в обратном порядке (по убыванию).
- 3) Удалить запись. Пользователь вводит ID записи, которую необходимо удалить. Программа находит данную запись и заменяет её ID на -1 для

«отмены» отображения в функциях вывода. Поля при этом не заменяются.

- 4) Изменить запись. Пользователь вводит IDзаписи, которую необходимо изменить. Программа находит запись с данным IDи предлагает пользователю ввести новые данные для экземпляра. Пользователь вводит новые данные, и программа сохраняет новый экземпляр в базе данных.
- 5) Загрузить БД в файл. Пользователь вводит название файла, в который необходимо записать данные. Если данный файл уже существует, то данные в нем перезаписываются, в противном случае создается новый файл, в который записывается база данных.
- 6) Загрузить БД из файла. Пользователь вводит название файла, из которого необходимо считать данные. Если данный файл не существует или не соответствует формату, то программа выводит ошибку, иначе считывает базу данных из файла. Если на момент считывания базы данных в программе находились экземпляры другого рода, то они затираются новыми.
- 7) Выход. Выход из программы.

Навигация в меню осуществляется с помощью клавиатуры, вводом цифр от 0 до 6.

Описание разработки программы

Разработанная программа состоит из нескольких модулей, которые реализовывались в следующем порядке:

- 1) Data_Base.cpp – основной файл программы. Меню программы.
- 2) constst.cpp – файл с константами.
- 3) add_provider.cpp – добавление нового элемента в массив (пользовательская оболочка).
- 4) init.cpp – для инициализации односвязного динамического списка структур.
- 5) append.cpp – добавление нового элемента в список.
- 6) change_example.cpp – изменениеэлемента списка.
- 7) del_example.cpp – удалениеэлемента из списка.
- 8) print_database.cpp – выводзаписей в определенном формате.
- 9) read_form_file.cpp – чтениебазыданных из файла.
- 10) write_to_file.cpp – записьбазыданных в файл.
- 11) answer_handler.cpp – корректнаяконвертация ответов Y/N, Д/Н.
- 12) color_print.cpp – цветной вывод сообщений в консоль.
- 13) crop_string.cpp – отделение знака ‘\n’ от строки после ввода.

По порядку реализации модулей можно понять, что использовался нисходящей подход реализации программы. Этот метод основан на том, что

сначала проектируются основные компоненты программы, а затем уже дорабатываются ее мелкие детали. Это позволяет упростить ее тестирование, так как каждый модуль тестируется сразу же после его реализации, что помогает в создании более структурированных программ.

Отладка и тестирование

В качестве среды разработки была выбрана программа Microsoft Visual Studio 2019. Программа обладает всеми средствами необходимыми при разработке и отладке программы. Для отладки использовались несколько возможностей Visual Studio: точка останова, трассировка, анализ содержимого переменных.

Тестирование проводилось во время разработки и также после завершения разработки. В ходе нее было выявлено огромное количество проблем, связанных с работой с файлами, работой с памятью, работой с полями структуры.

Описание программы

Приложение Data_Base.exe является основным модулем программы. При запуске программы происходит вывод заставки-приветствия, после чего пользователю необходимо выбрать пункт меню для дальнейшего использования базы данных. Выход из меню осуществляется после выбора пользователем нужного пункта. Описание состояний программы выполнено в таблице ниже.

Клавиши, вызывающее событие	Действие пользователя	Действие программы
1,Enter	Выбран пункт «Добавить запись»	Запускается диалог добавления записи и добавление записи в БД.
2,Enter	Выбран пункт «Вывести записи»	Открывается новое меню см. Таблица 2.
3,Enter	Выбран пункт «Удалить запись»	Запускается диалог удаления записи и удаление записи из БД.

4, Enter	Выбран пункт «Изменить запись»	Запускается диалог изменения записи и изменение записи из БД.
5, Enter	Выбран пункт «Загрузить БД в файл»	Запускается диалог и запись существующей БД в файл.
6, Enter	Выбран пункт «Загрузить БД из файла»	Запускается диалог и загрузка БД из файла.
0, Enter	Выбран пункт «Выход»	Выполняется закрытие программы

Таблица 1.

Клавиши, вызывающее событие	Действие пользователя	Действие программы
1,Enter	Выбран пункт «Показать все записи»	На экран выводятся все записи находящиеся в базе данных
2,Enter	Выбран пункт «Поиск по фильтру»	Открывается новое меню см. Таблица 3.
3,Enter	Выбран пункт «Отсортировать записи»	Открывается новое меню см. Таблица 4.
0, Enter	Выбран пункт «Выход»	Выполняется выход в главное меню.

Таблица 2. – Работа функции print_database

Клавиши, вызывающее событие	Действие пользователя	Действие программы
1,Enter	Выбран пункт «Поиск по ID»	Программа выводит запись с IDуказанным пользователем.

2,Enter	Выбран пункт «Поиск по оператору»	Программа считывает наименование оператора связи и выводит подходящие записи
3,Enter	Выбран пункт «Поиск по тарифу»	Программа считывает название тарифа и выводит подходящие записи
4, Enter	Выбран пункт «Поиск по стоимости»	Программа выводит записи в указанном диапазоне стоимости услуг
5, Enter	Выбран пункт «Поиск по дате»	Программа считывает дату и выводит подходящие записи.
6, Enter	Выбран пункт «Поиск по доступности»	Программа принимает значение «Y»или «N» и выводит записи с указанным параметром доступности
0, Enter	Выбран пункт «Выход»	Выполняется выход в главное меню

Таблица 3 – Поиск по фильтру

Клавиши, вызывающее событие	Действие пользователя	Действие программы
1,Enter	Выбран пункт «Сортировать в обратном порядке по ID»	На экран выводятся все записи в обратном порядке (с конца к началу)
2,Enter	Выбран пункт «Сортировать по цене»	Записи сортируются по цене от меньшего к большему и выводятся на экран
3,Enter	Выбран пункт «Сортировать по ценеобратном порядке»	Записи сортируются по цене от большего к меньшему и выводятся на экран

Таблица 4 – Сортировка записей

Далее на рисунке 1 представлена схема взаимодействия программы.

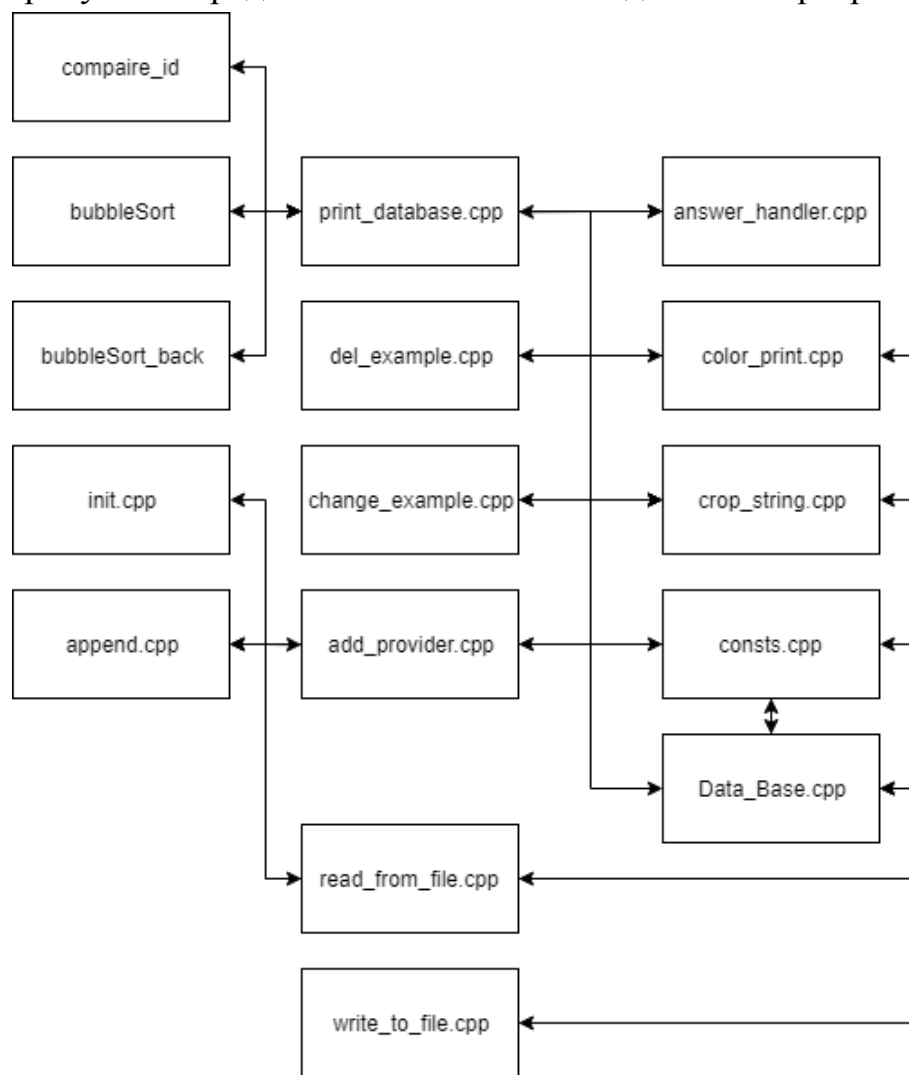


Рисунок - схема взаимодействия программы

Ниже, на рисунке 2 представлена схема данных.

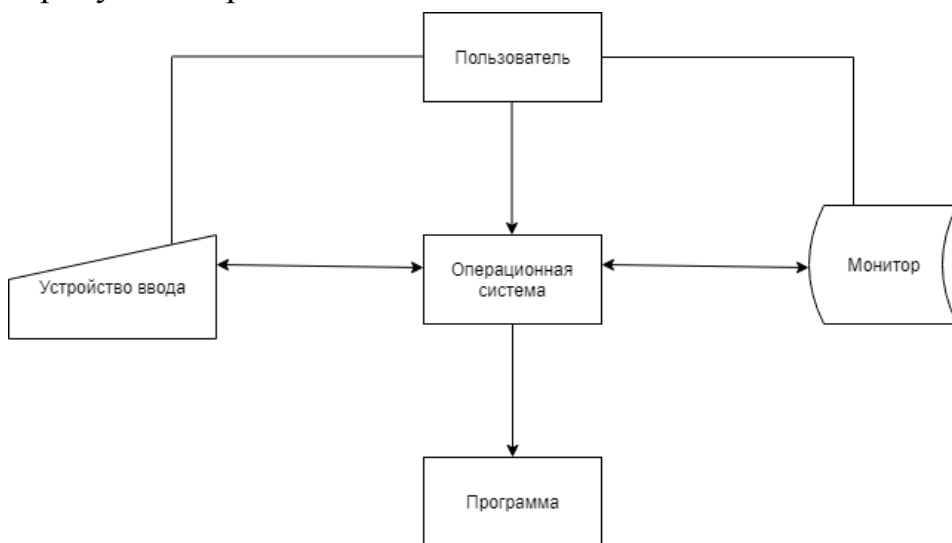


Рисунок - схема данных

Схема программы. Функция Main

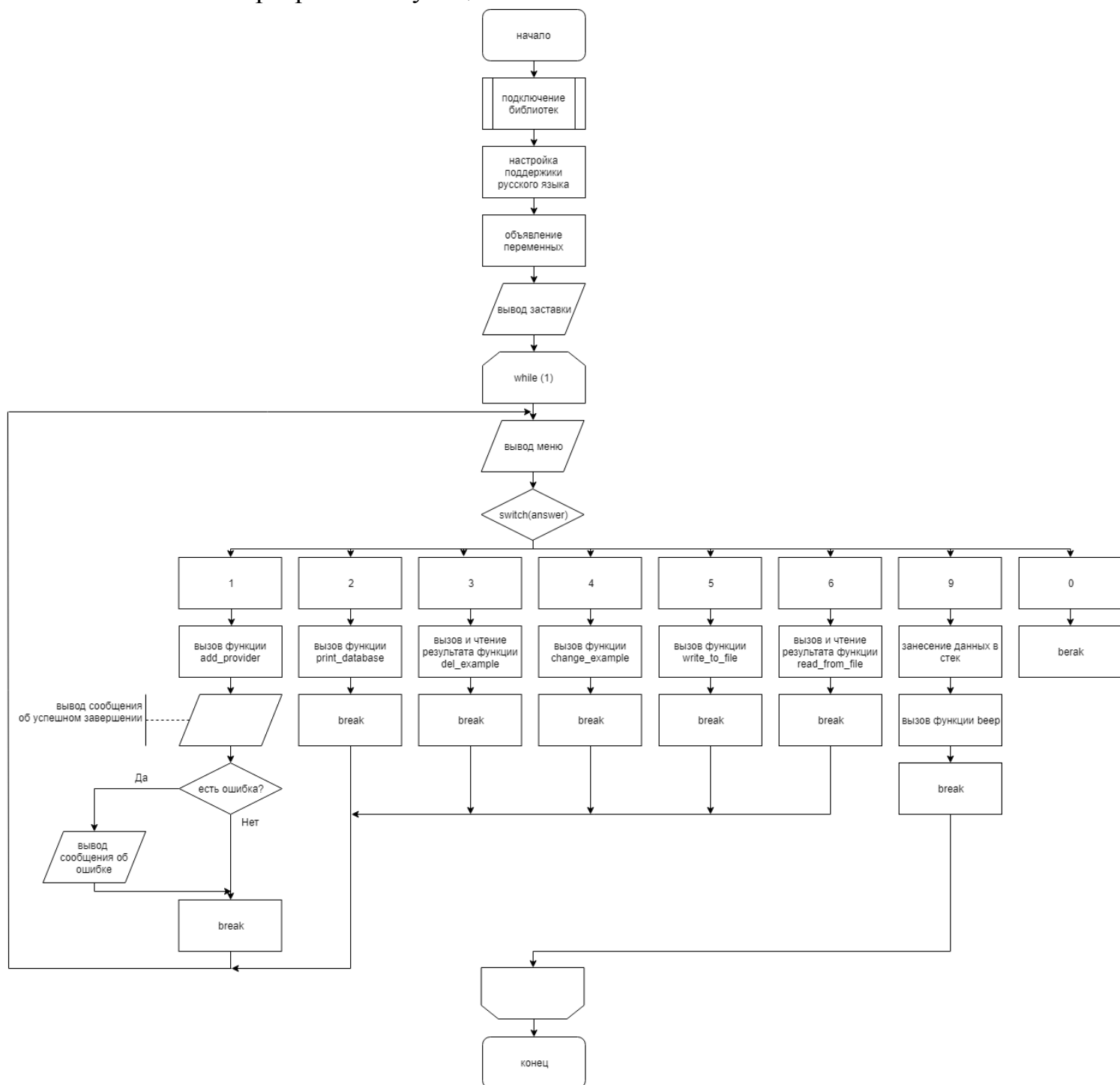
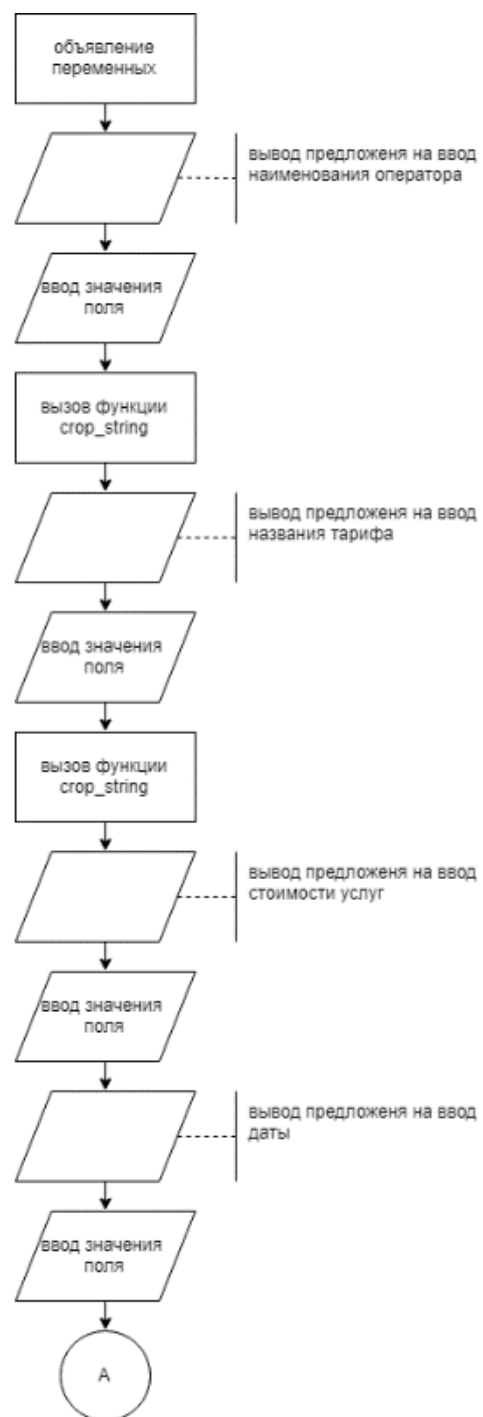
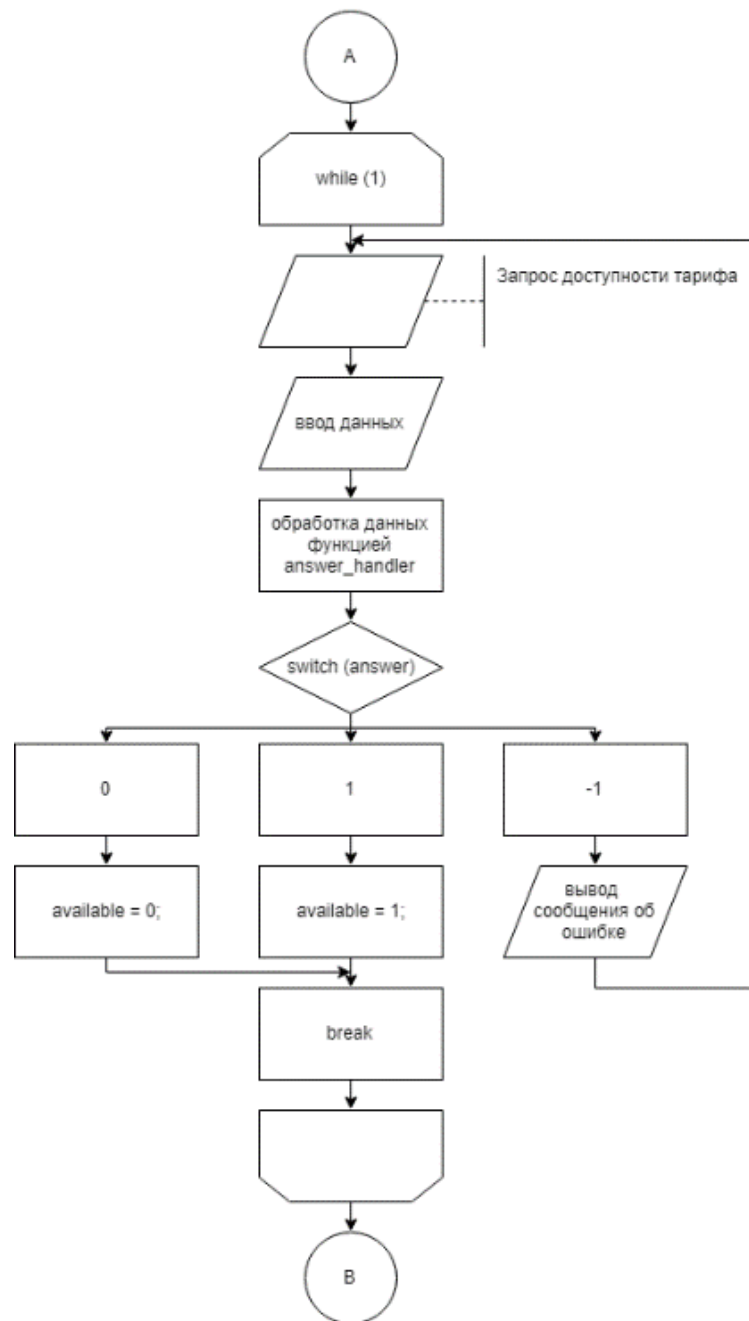


Рисунок - функция main

Схема программы. Функция add_provider





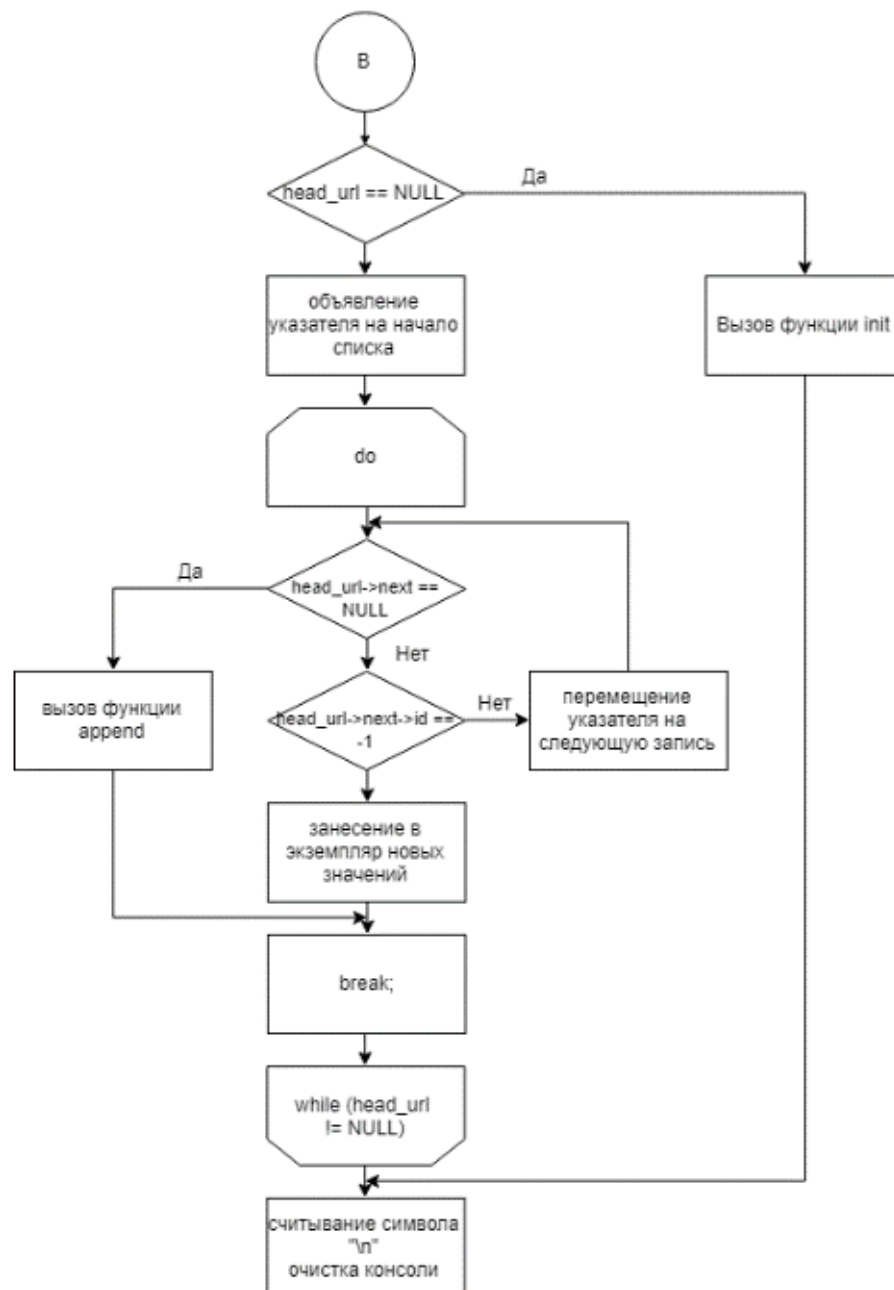
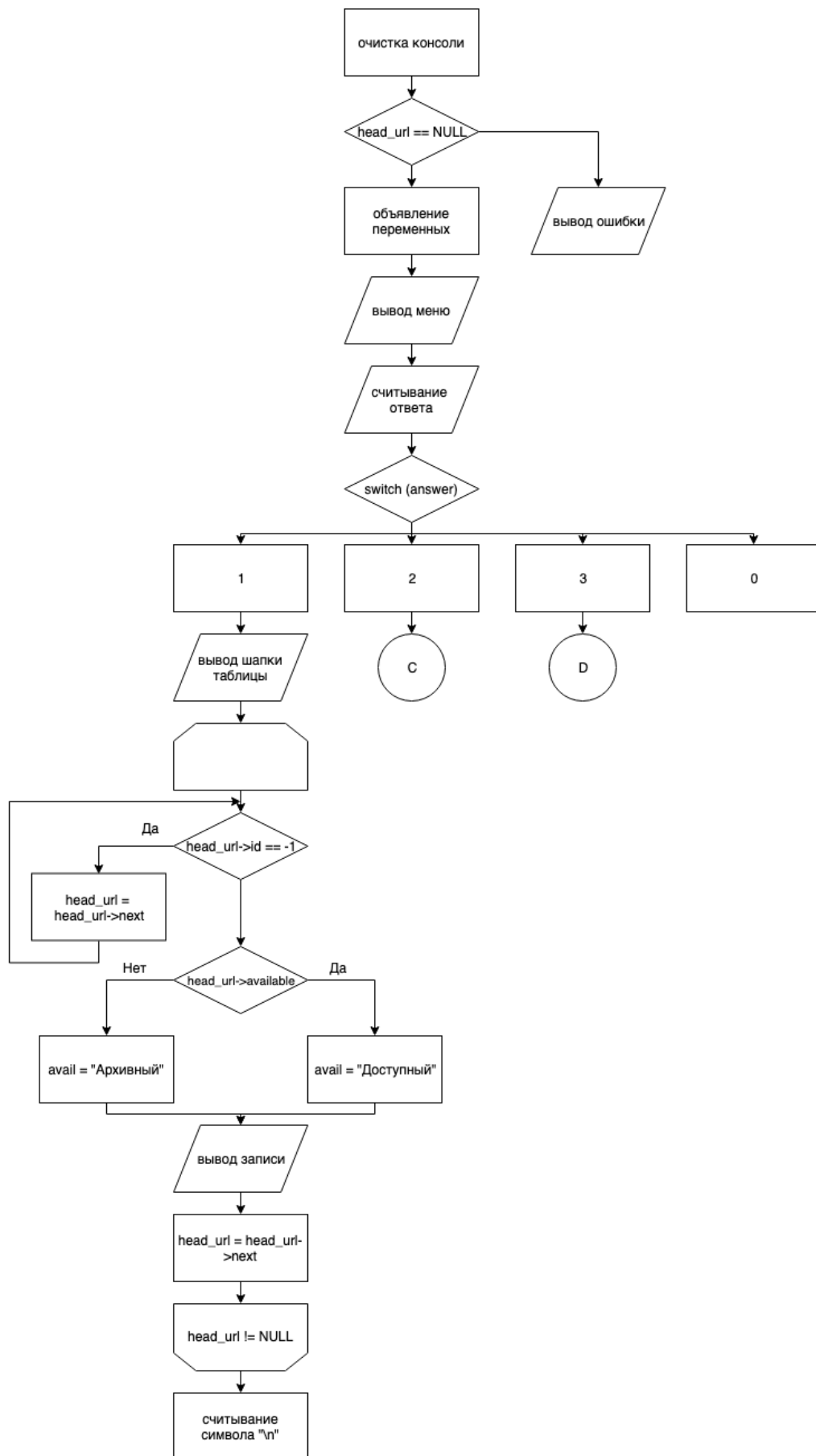
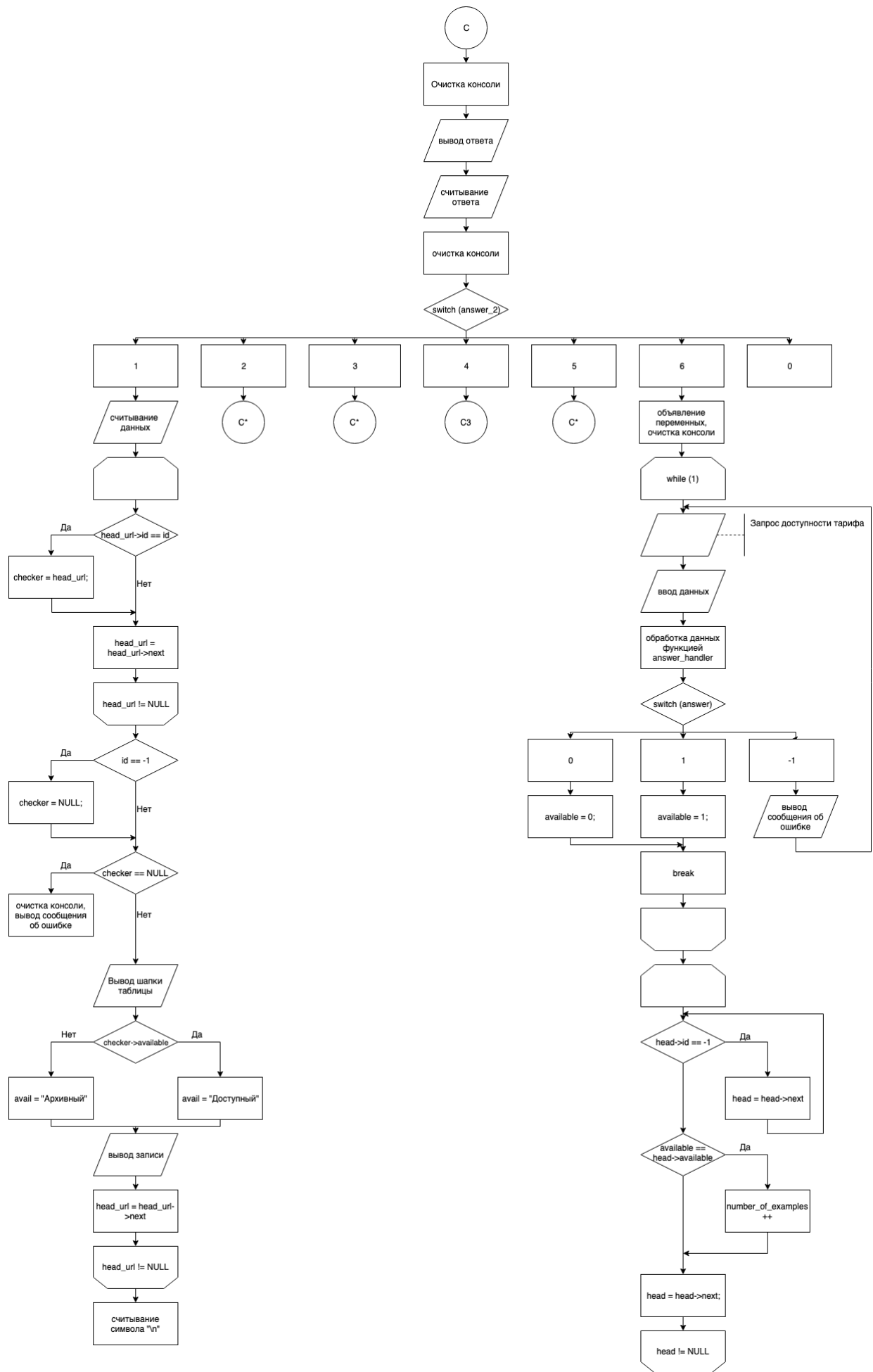
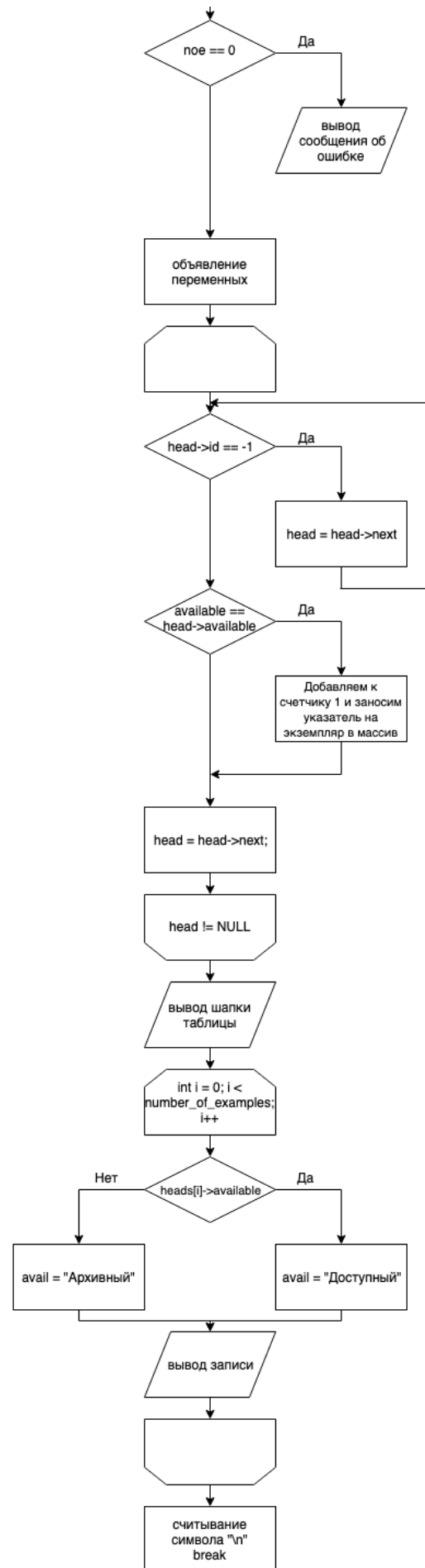


Рисунок - Функция add_provider

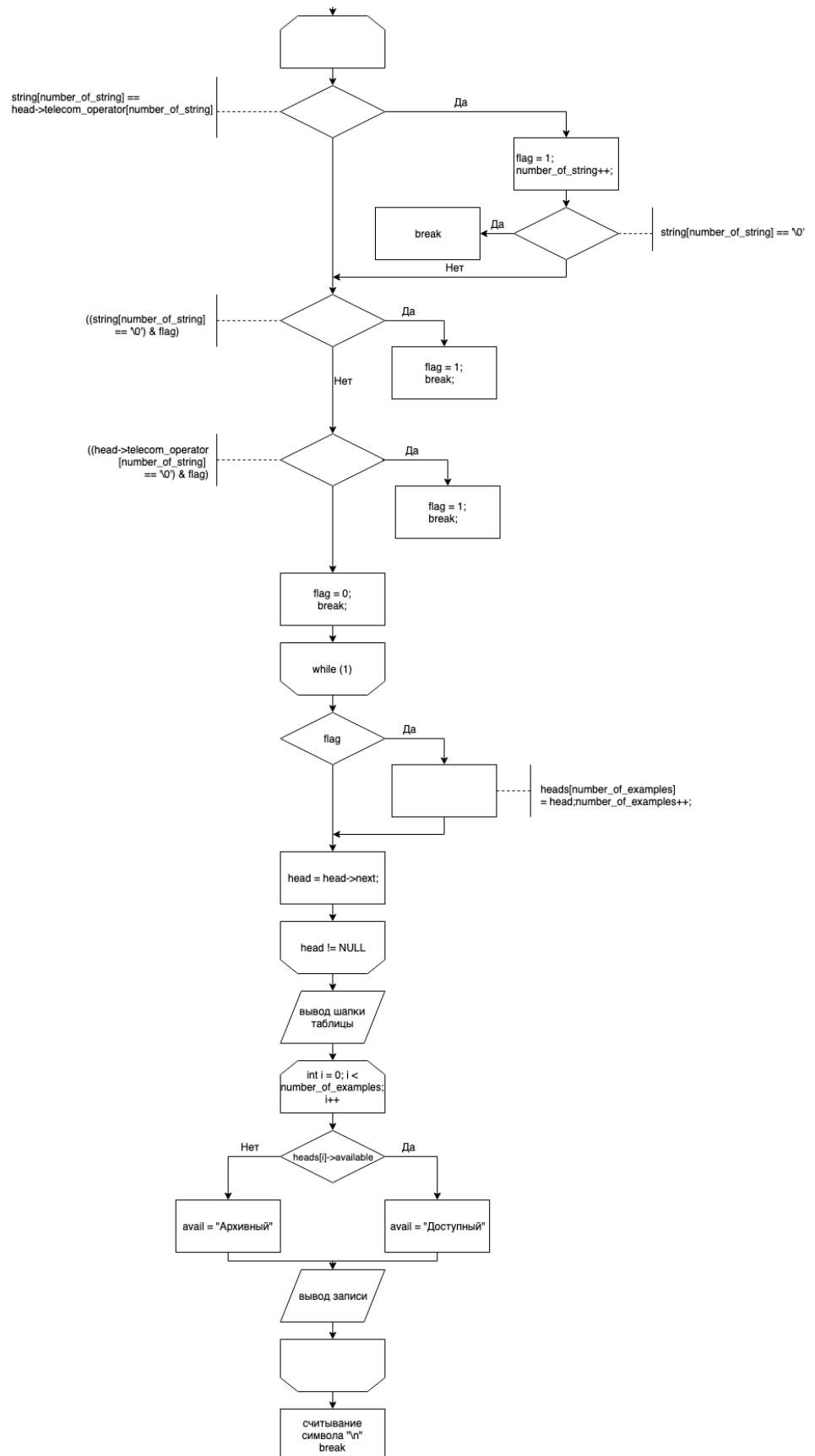
Схема программы. Функция print_database

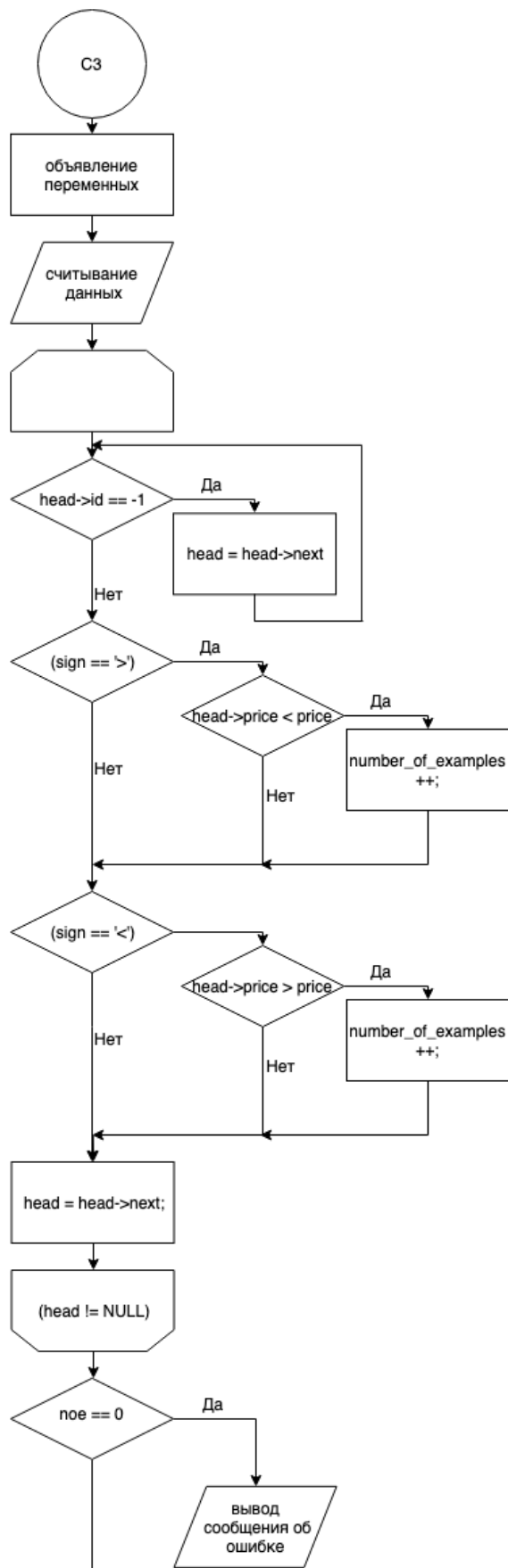


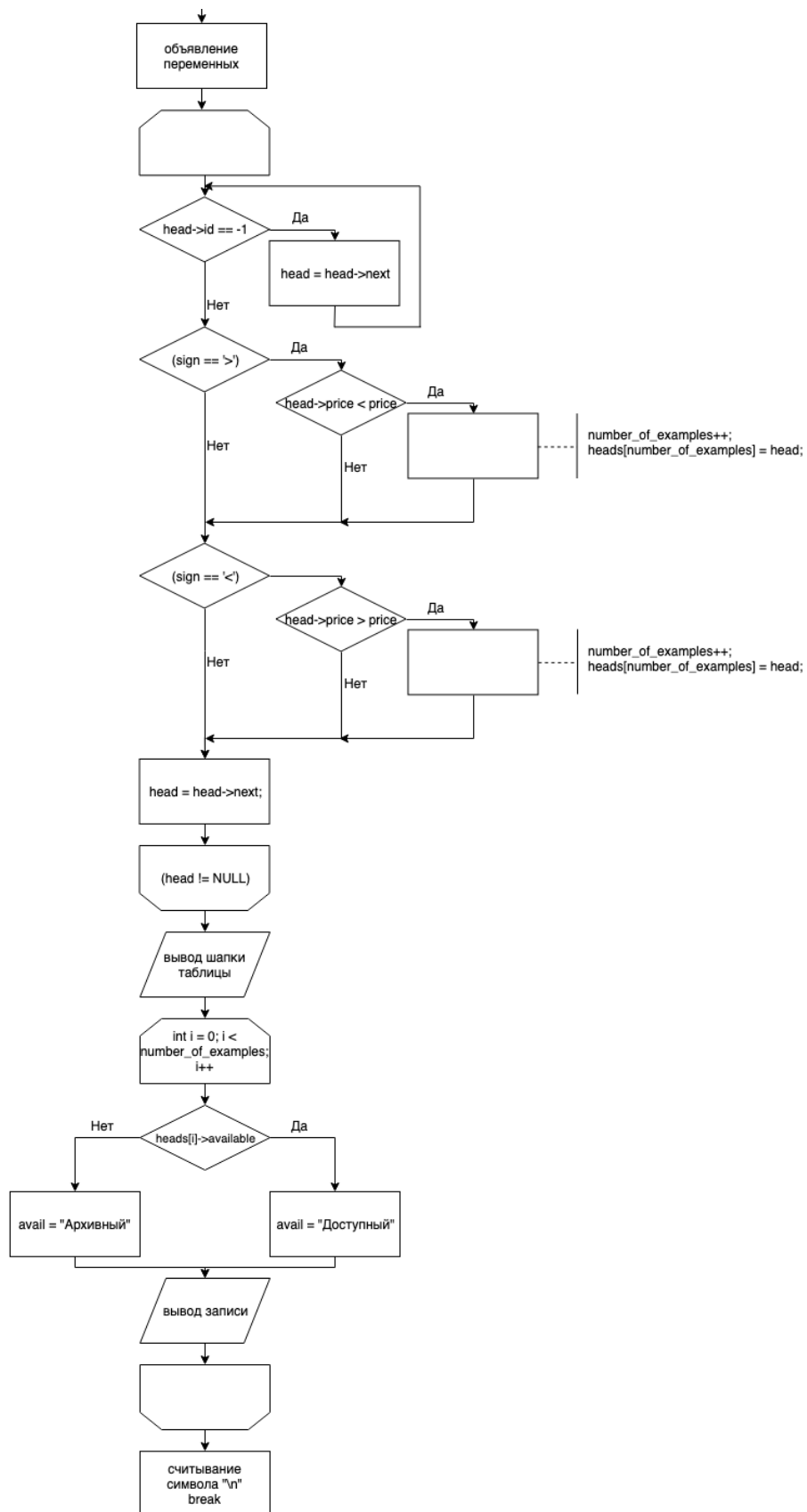












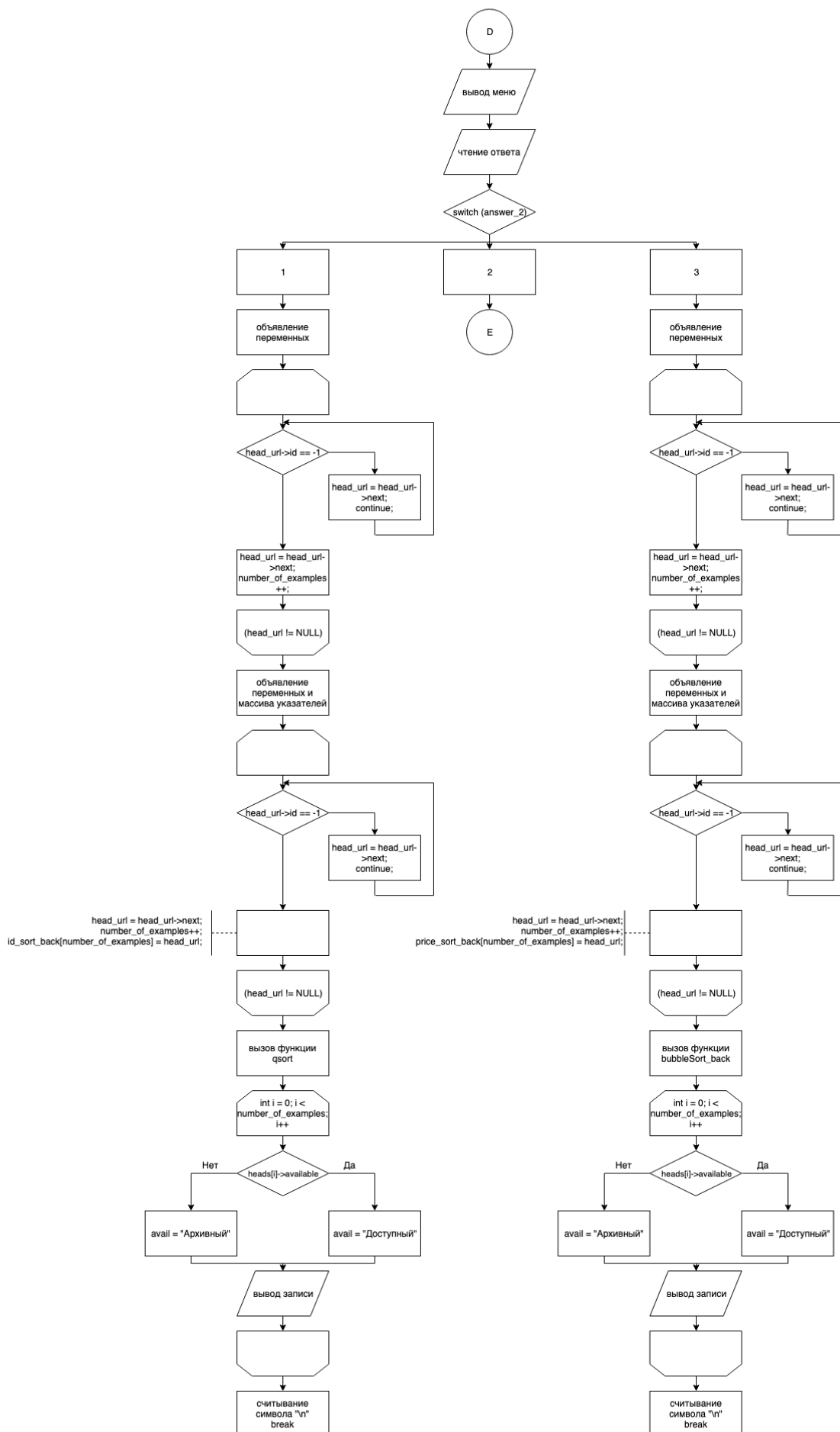


Рисунок - функция print_database

Ниже приведена иерархическая схема программы.

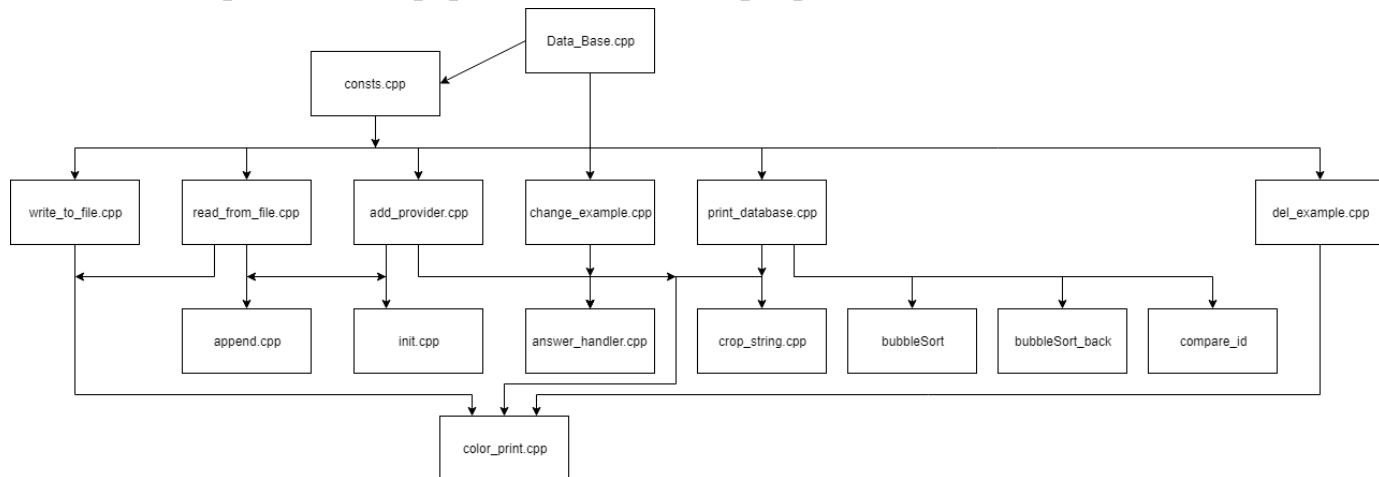
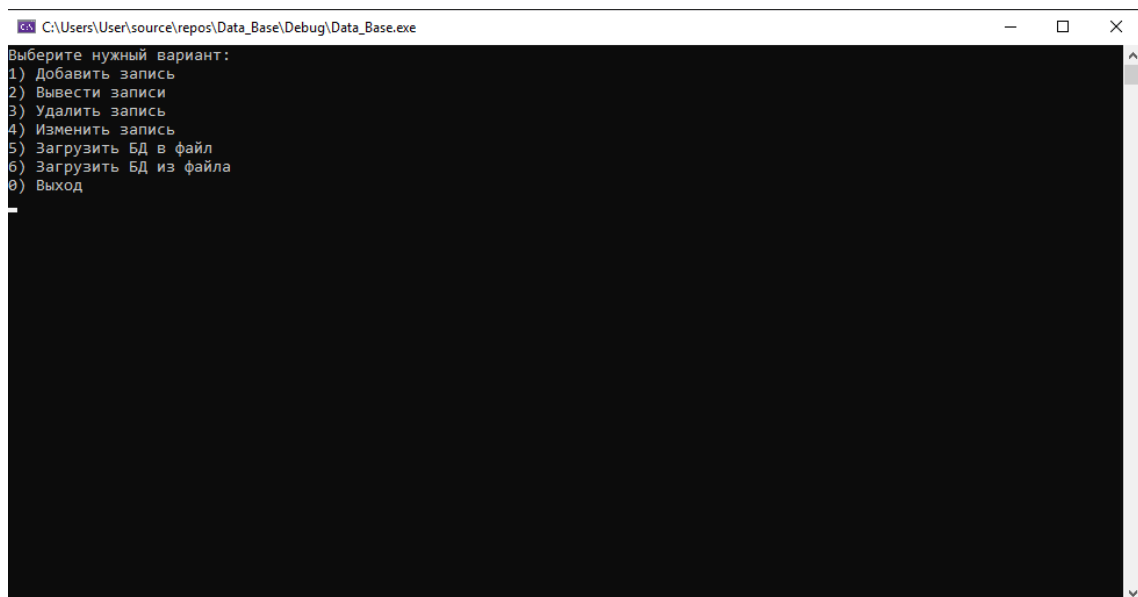


Рисунок - иерархическая схема программы

Руководство пользователя

Программа Data_Base.exe предназначена для хранения информации об операторах сотовой связи их тарифах и стоимости услуг. Программа имеет интуитивно понятный интерфейс и поддерживает такие операции как добавление новой записи в базу данных, удаление записи из базы данных, изменение записи в базе данных, поиск записи в базе данных, сохранение существующей базы данных в файл, импорт базы данных из файла, сортированный вывод записей базы данных и фильтрация записей в базе данных.

Для открытия и вывода на экран существующего каталога необходимо выбрать пункт меню «Вывести записи». В открывшемся меню выбрать пункт «Показать все записи».



The image shows two screenshots of a Windows command prompt window titled "C:\Users\User\source\repos\Data_Base\Debug\Data_Base.exe".

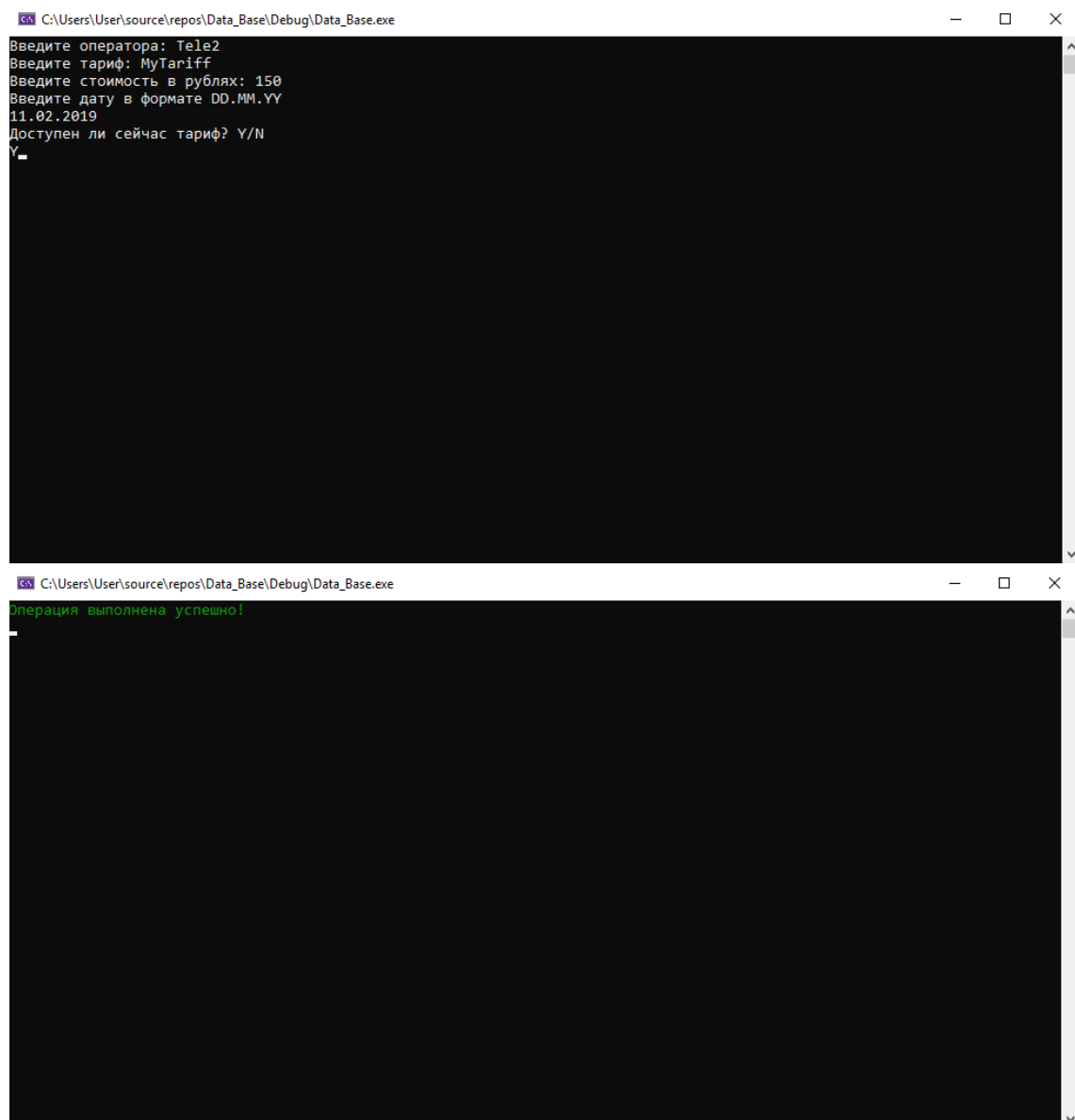
The top screenshot displays a menu with the following options:

```
1) Показать все записи
2) Поиск по фильтру
3) Отсортировать записи
0) Выход
```

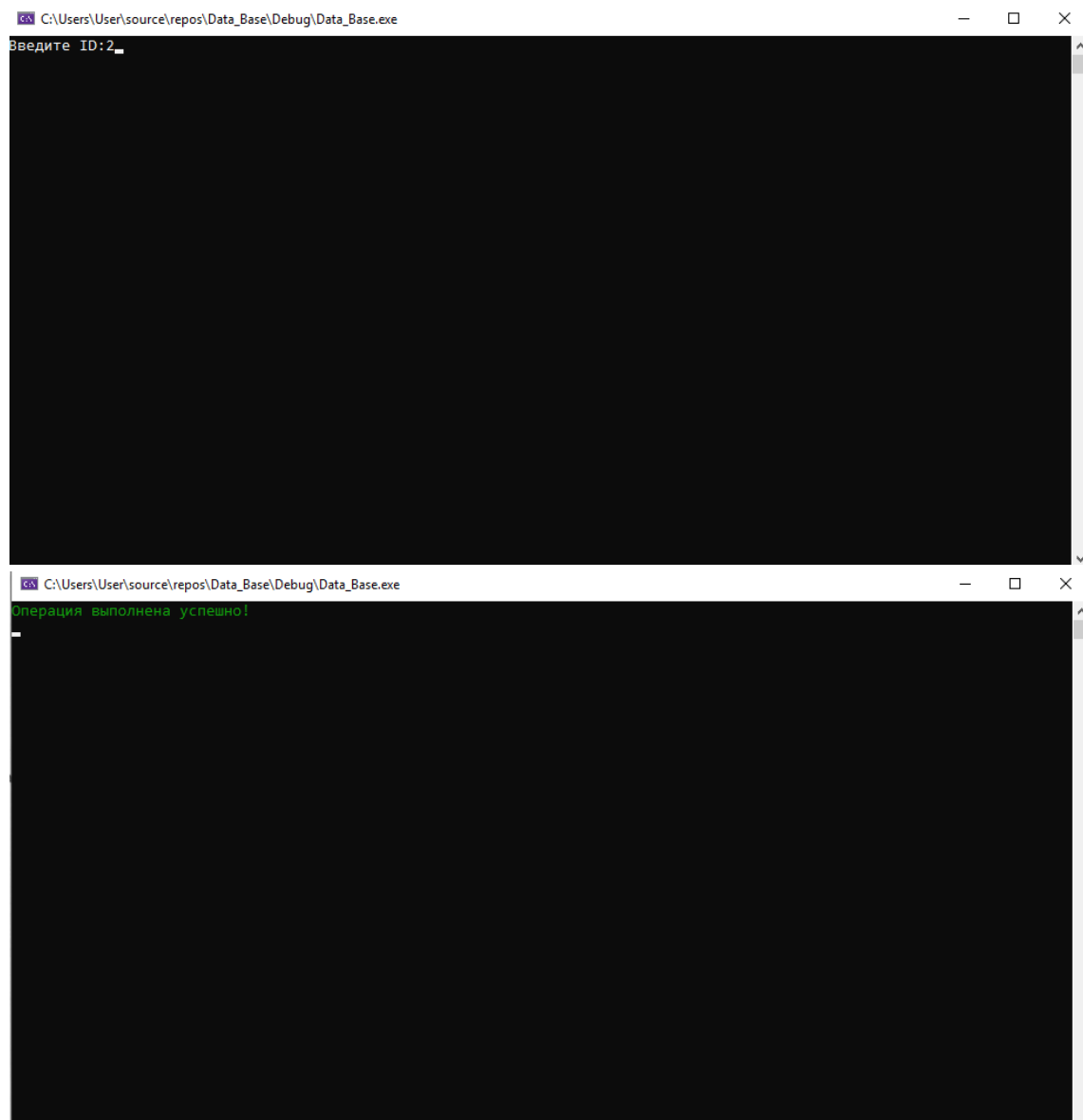
The bottom screenshot displays a table of data with the following columns: id, оператор, тариф, стоимость, дата, and доступность. The data is as follows:

id	оператор	тариф	стоимость	дата	доступность
0	Tele2	Tele2	600	11.02.2019	Архивный
1	MegaPhone	Mega	230	11.05.2018	Архивный
2	Beeline	Bee	750	11.06.2014	Архивный
3	Tele2	All	140	11.06.2005	Доступен

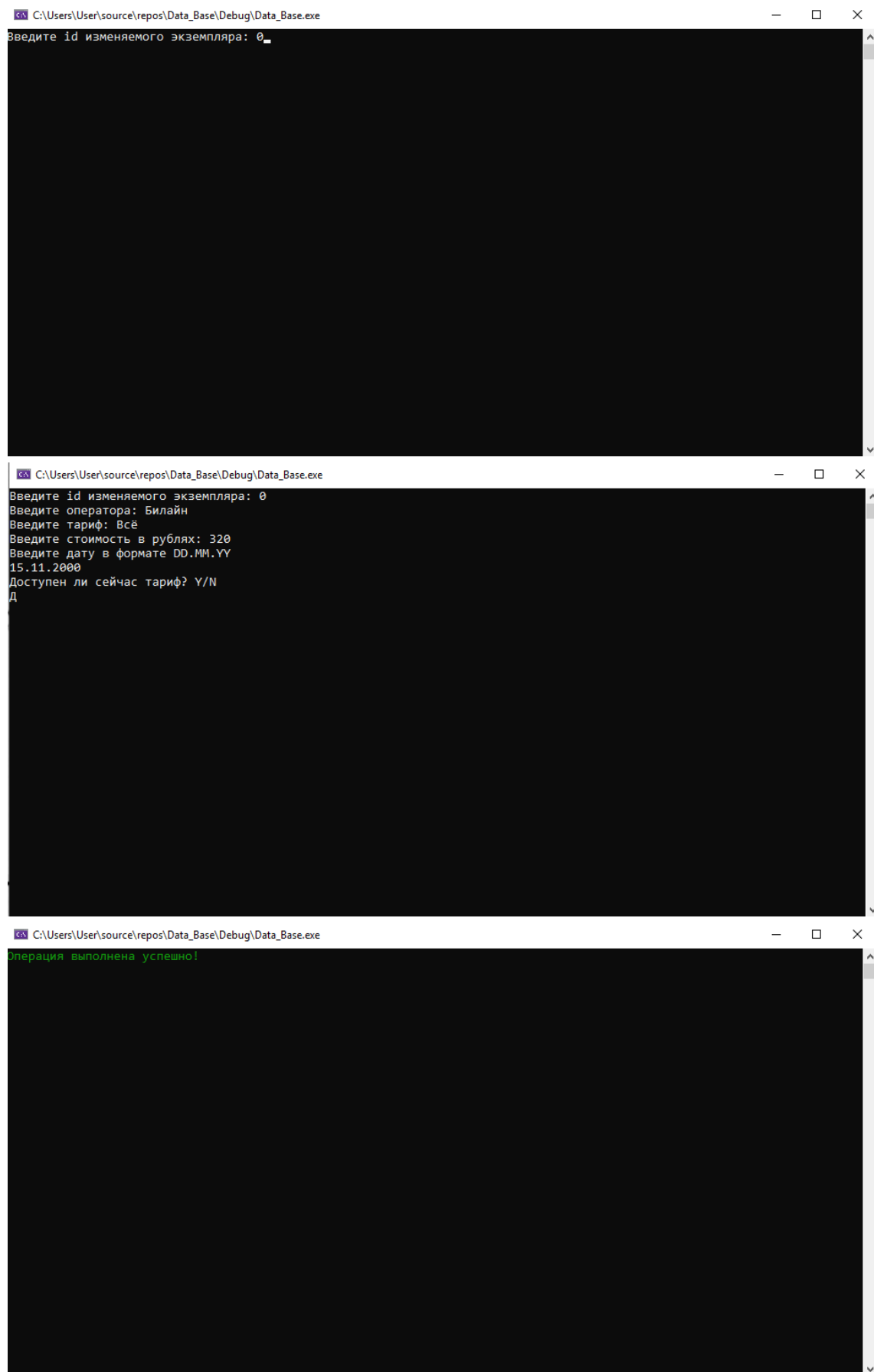
Для добавления записи в каталог необходимо выбрать пункт меню «Добавить запись». Далее необходимо заполнить форму структуры из пяти пунктов: ввести наименование оператора, нажать клавишу Enter, ввести название тарифного плана, нажать клавишу Enter, ввести стоимость услуг, нажать клавишу Enter, ввести дату выпуска тарифа, нажать клавишу Enter, ввести доступность тарифа, нажать клавишу Enter. В случае если не возникли ошибки на экране появится надпись «Операция выполнена успешно!».



Для удаления записи из каталога необходимо выбрать пункт меню «Удалить запись». Далее необходимо ввести ID удаляемой записи и нажать клавишу Enter. В случае если не возникли ошибки на экране появится надпись «Операция выполнена успешно!».

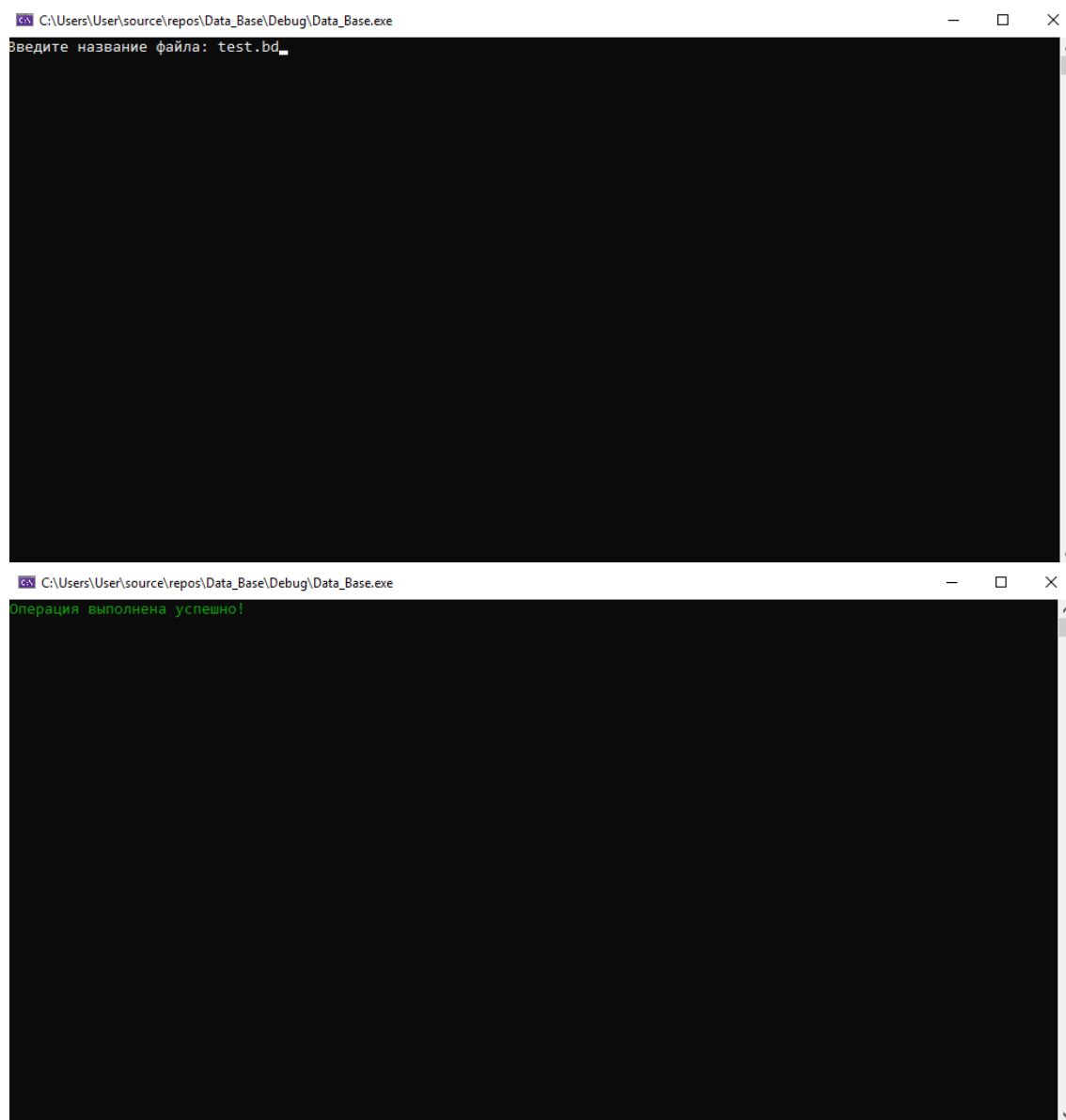


Для изменения записи необходимо выбрать пункт «Изменить запись» в меню. Далее необходимо ввести ID изменяемой записи и ввести новые данные для экземпляра из пяти пунктов: ввести наименование оператора, нажать клавишу Enter, ввести название тарифного плана, нажать клавишу Enter, ввести стоимость услуг, нажать клавишу Enter, ввести дату выпуска тарифа, нажать клавишу Enter, ввести доступность тарифа, нажать клавишу Enter. В случае если не возникли ошибки на экране появится надпись «Операция выполнена успешно!».

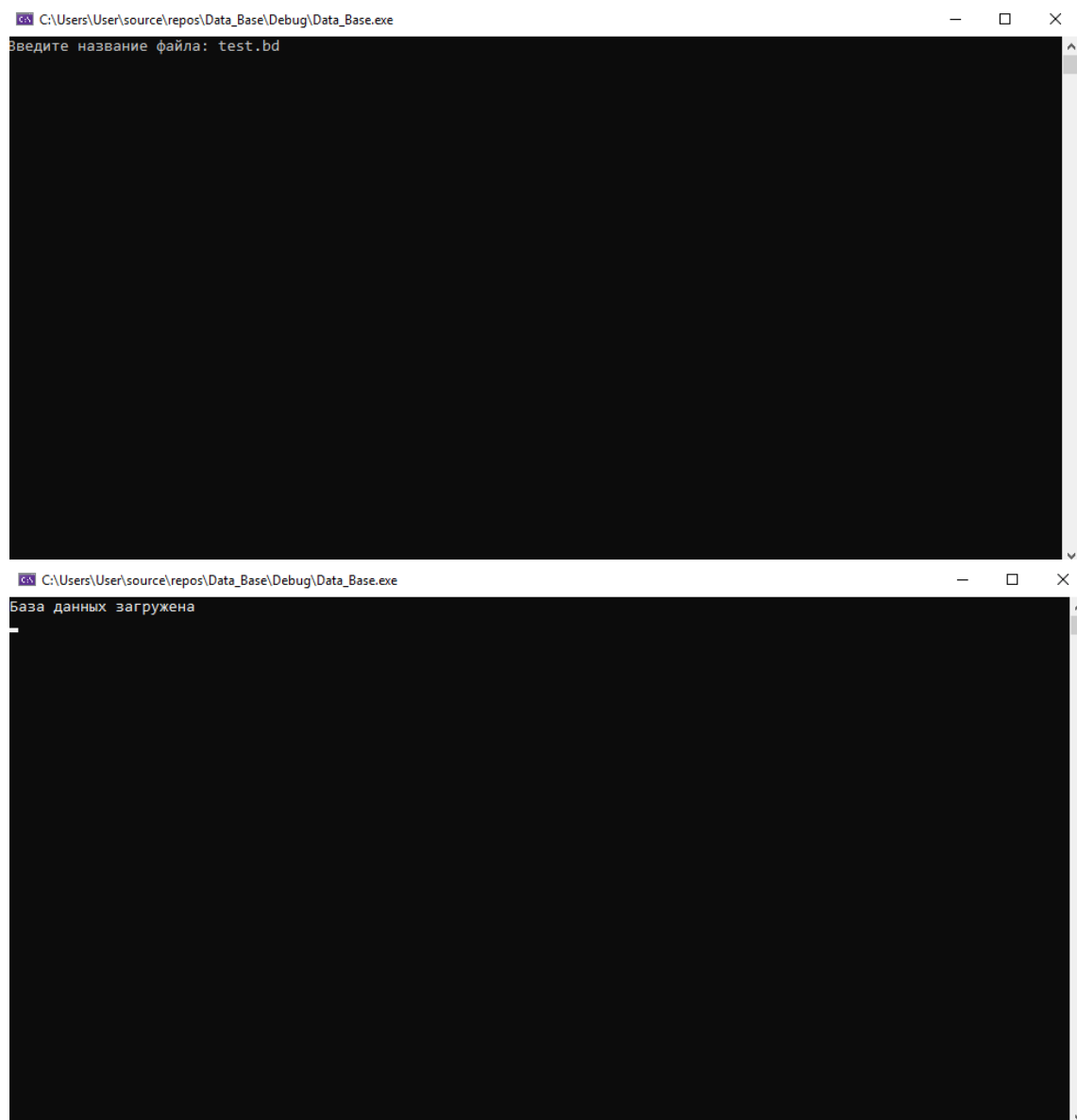


Для того чтобы сохранить текущую базу данных в файл необходимо выбрать пункт меню «Загрузить БД в файл». Далее ввести имя файла, в

который будет записана база данных. Если файл уже существует, то программа удалит информацию из файла и запишет в него каталог. В случае если не возникли ошибки на экране появится надпись «Операция выполнена успешно!».



Для того чтобы импортировать базу данных из файла необходимо выбрать пункт меню «Загрузить БД из файла». Далее ввести имя файла, из которого произойдет чтение данных, нажать клавишу Enter. Текущая база данных будет заменена на новую, из файла. В случае ошибки программа выдаст сообщение «Ошибка при чтении из файла». В случае если не возникли ошибки на экране появится надпись «База данных загружена».



Для того чтобы записи были выведены в отсортированном виде необходимо в главном меню выбрать пункт «Вывести записи». В открывшемся меню выбрать пункт «Отсортировать записи». Далее выбрать один из трех параметров сортировки: «Сортировать в обратном порядке по ID», «Сортировать по цене», «Сортировать по цене в обратном порядке», нажать клавишу Enter.

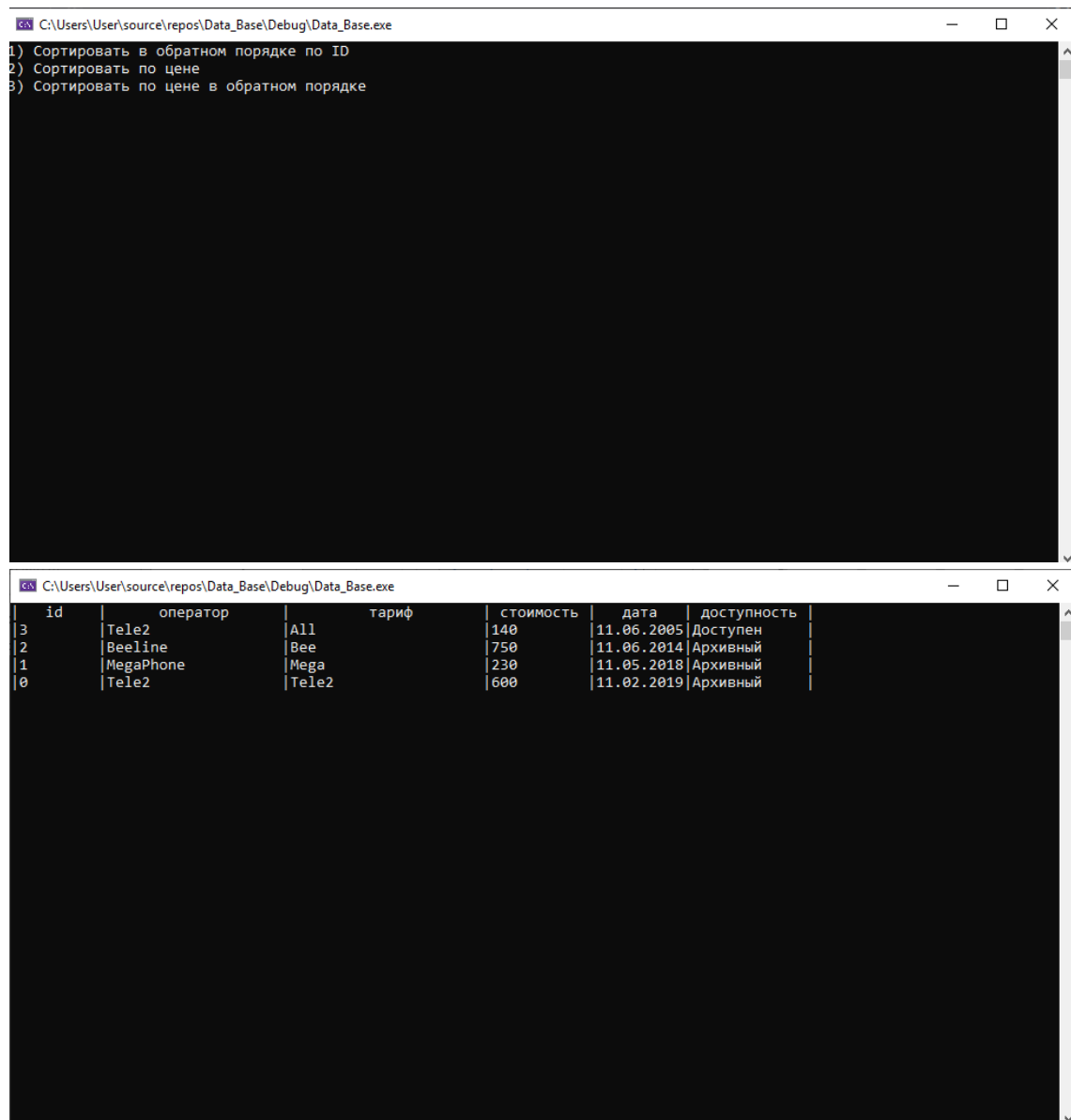


Рисунок - сортировка по ID в обратном порядке

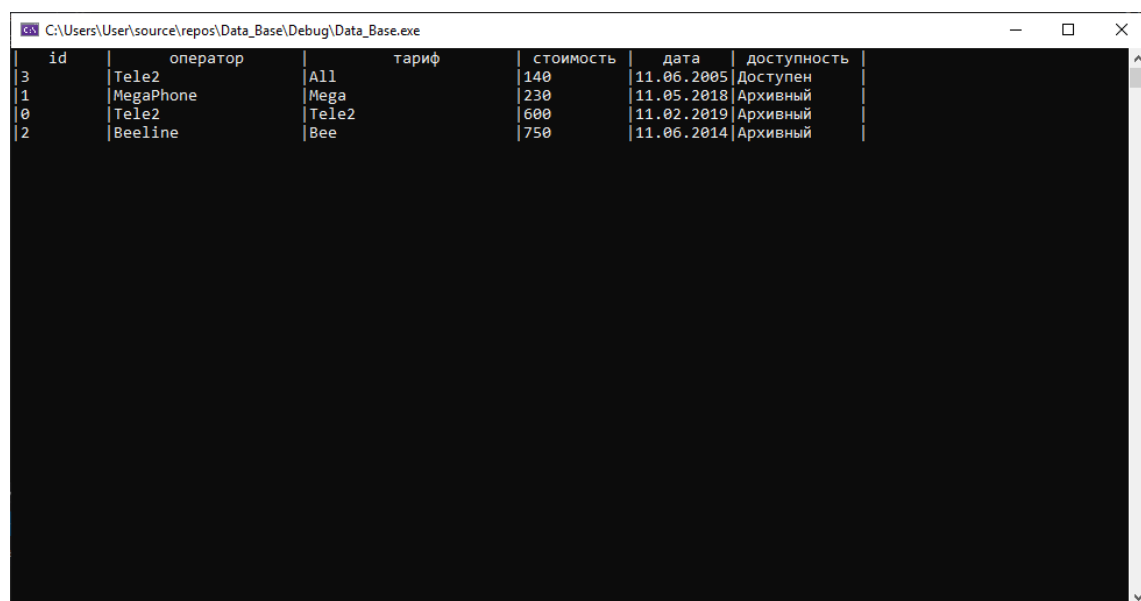
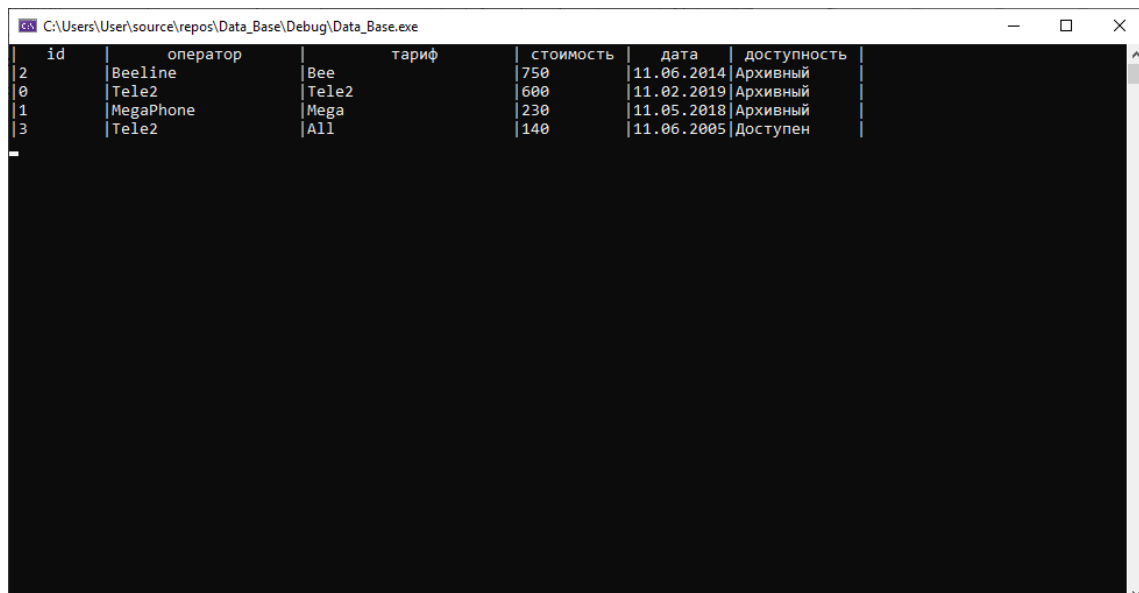


Рисунок - Сортировка по цене

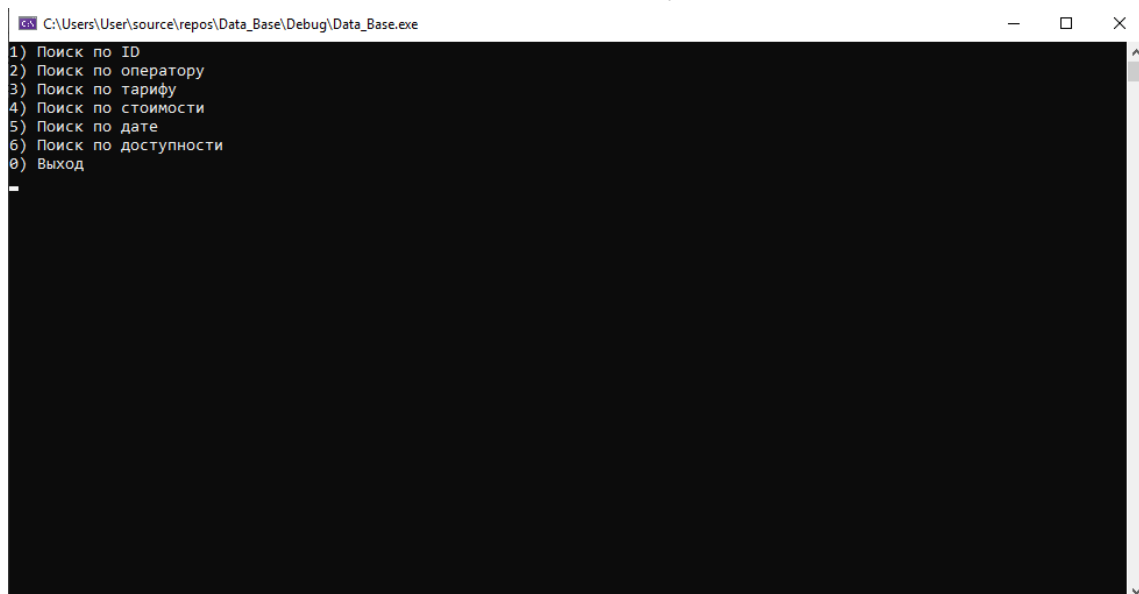


The screenshot shows a console window titled "C:\Users\User\source\repos\Data_Base\Debug\Data_Base.exe". It displays a table with the following data:

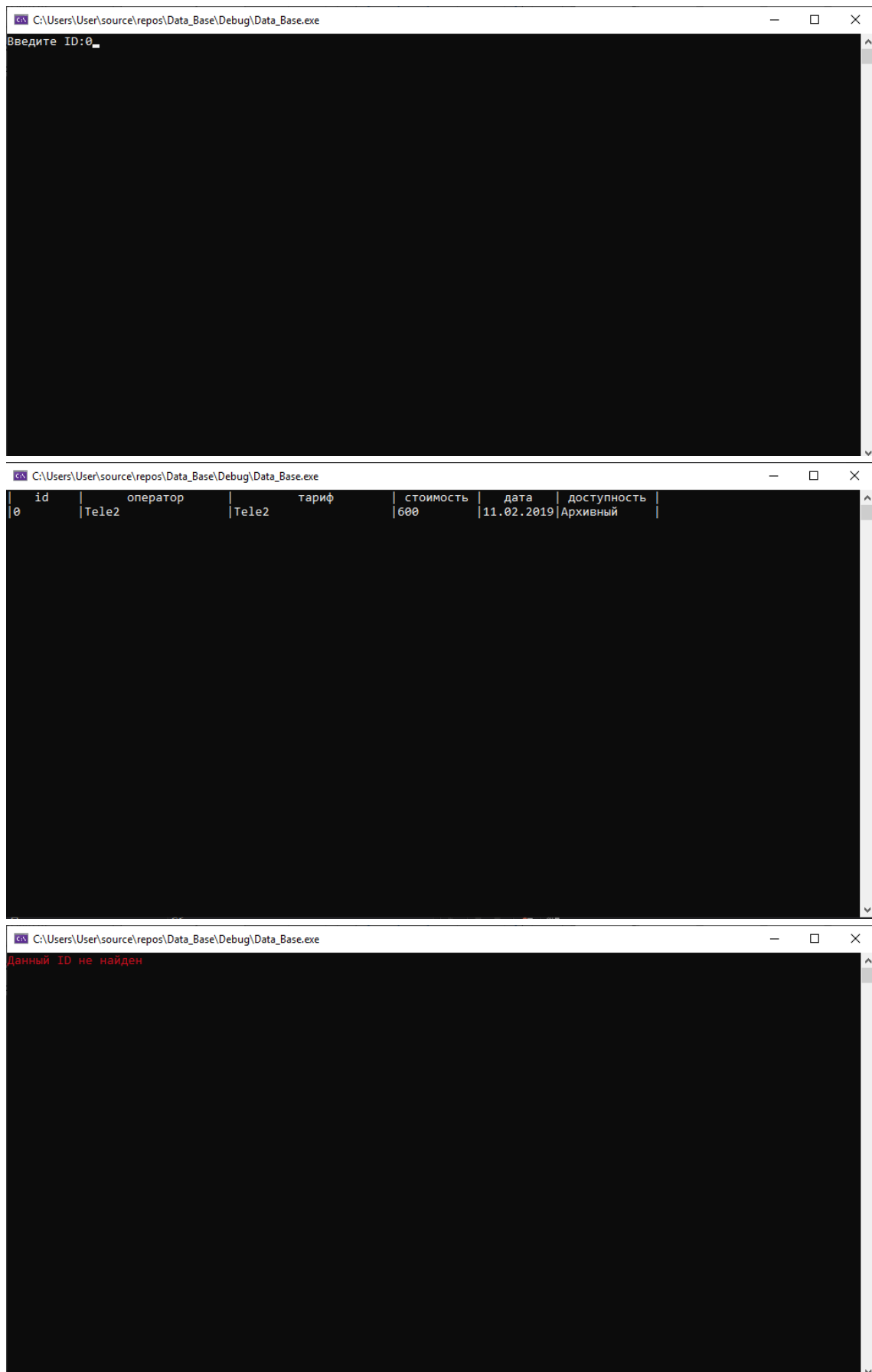
id	оператор	тариф	стоимость	дата	доступность
2	Beeline	Bee	750	11.06.2014	Архивный
0	Tele2	Tele2	600	11.02.2019	Архивный
1	MegaPhone	Mega	230	11.05.2018	Архивный
3	Tele2	All	140	11.06.2005	Доступен

Рисунок - Сортировка по цене в обратном порядке

Для того чтобы найти определенную запись или найти группу записей по критерию необходимо выбрать пункт меню «Вывести записи». В открывшемся меню выбрать пункт «Поиск по фильтру». Далее выбрать один из шести пунктов: «Поиск по ID», «Поиск по оператору», «Поиск по тарифу», «Поиск по стоимости», «Поиск по дате», «Поиск по доступности».

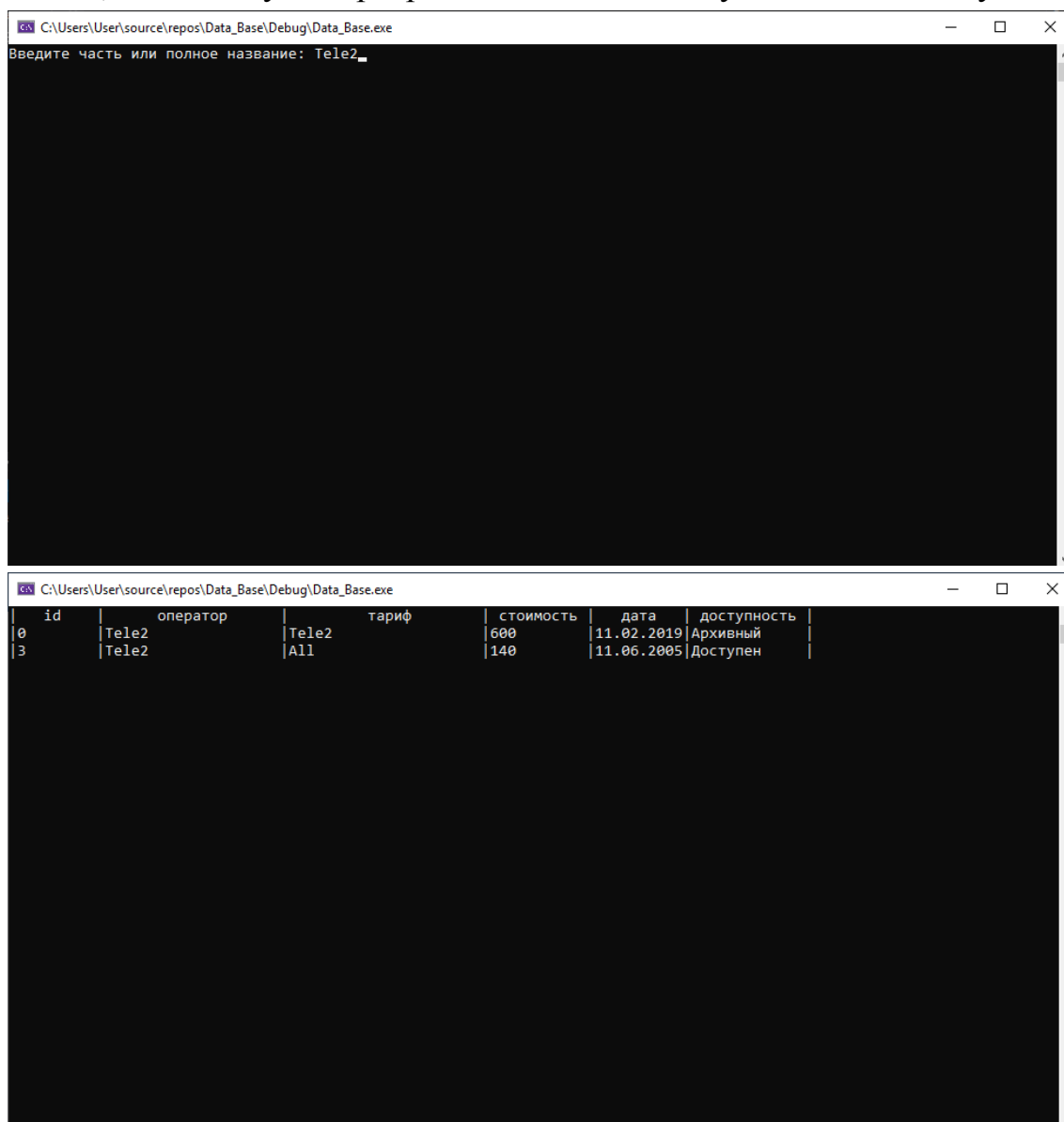


Поиск по ID предназначен для поиска конкретной записи. Пользователю следует ввести ID искомой записи, нажать клавишу Enter. В случае, если запись находится в базе данных, программа выведет её на экран, в противном случае выдаст ошибку «Данный ID не найден».

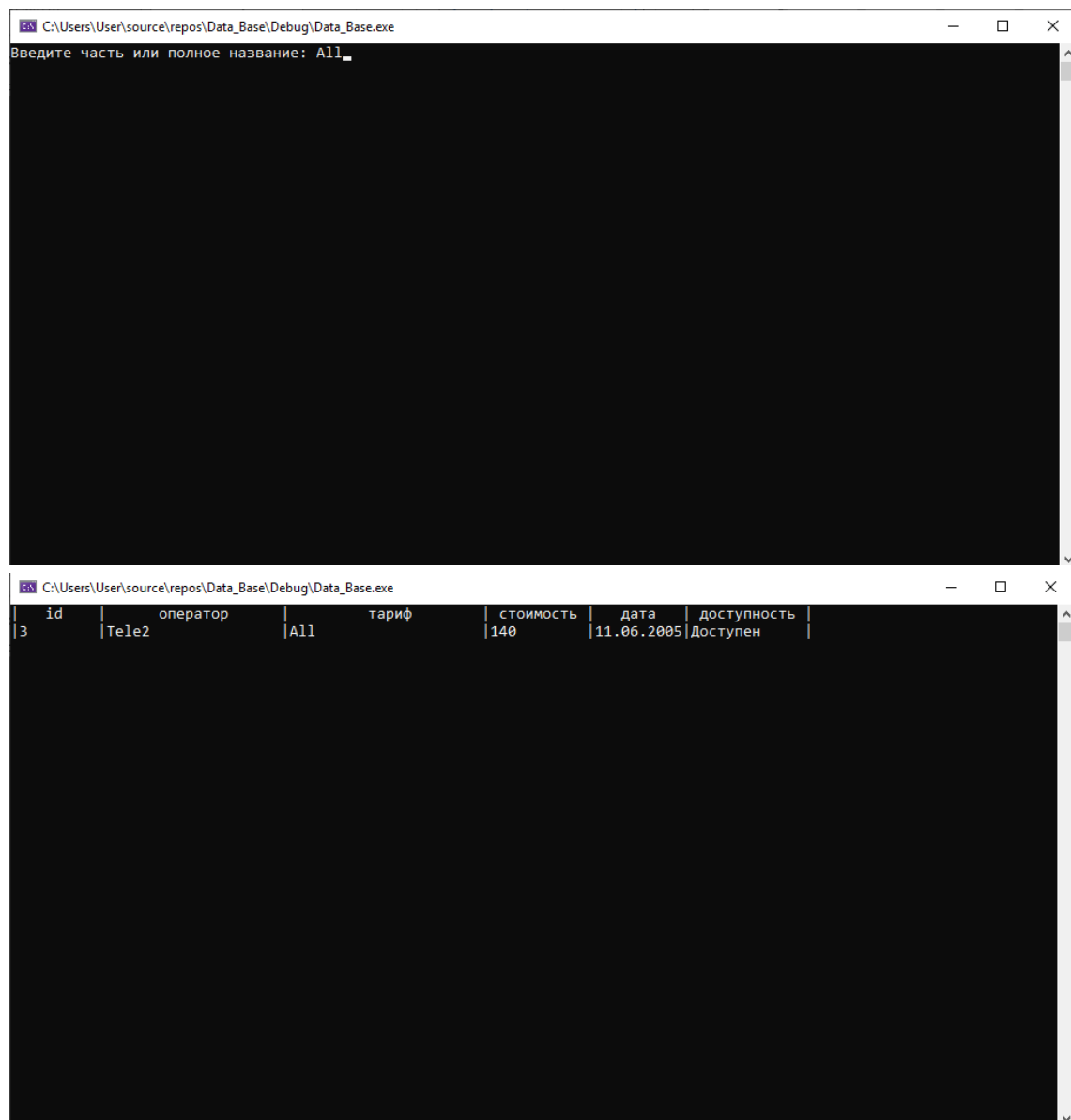


Поиск по оператору предназначен для поиска записей определенного оператора. Пользователь может ввести как полное название, так и его

начальную часть, далее нужно нажать клавишу Enter. Если записи с данным оператором есть в базе данных, то на экран выведется отфильтрованный список с записями, в ином случае программа выдаст ошибку «База данных пуста».



Поиск по тарифу необходим для фильтрации записей по тарифам. Пользователь может ввести как полное название, так и его начальную часть, далее нужно нажать клавишу Enter. Если записи с данным тарифом есть в базе данных, то на экран выведется отфильтрованный список с записями, в ином случае программа выдаст ошибку «База данных пуста».



Поиск по стоимости необходим для нахождения диапазона тарифных планов операторов с определенной стоимостью. Пользователю необходимо ввести стоимость тарифного плана, нажать клавишу Enter, выбрать критерий фильтрации (ввести знак «>» или «<»), нажать клавишу Enter. Поиск осуществляется в формате [цена] [знак] [стоимость тарифного плана]. Если записи с данным тарифом есть в базе данных, то на экран выведется отфильтрованный список с записями, в ином случае программа выдаст ошибку «База данных пуста».

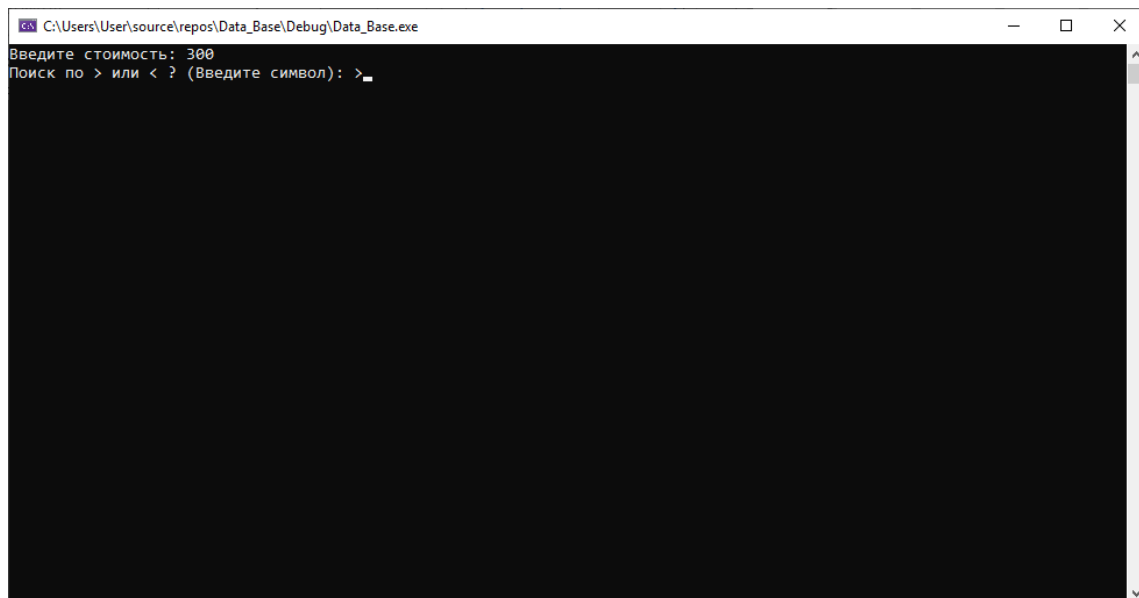


Рисунок - Сортировка по стоимости. Знак ">"

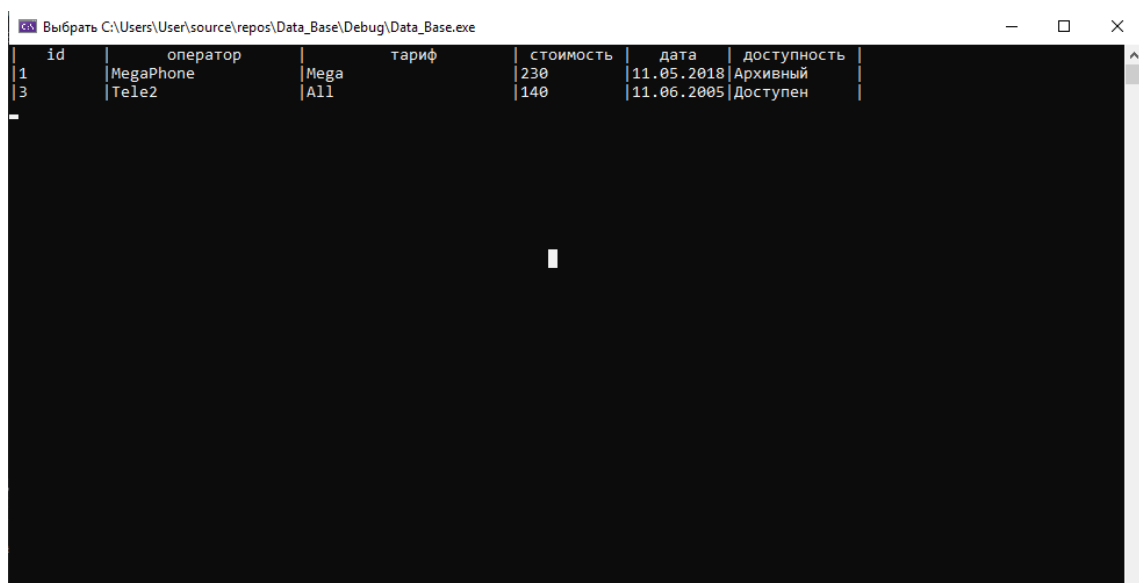


Рисунок - Результат сортировки по стоимости. Знак ">"

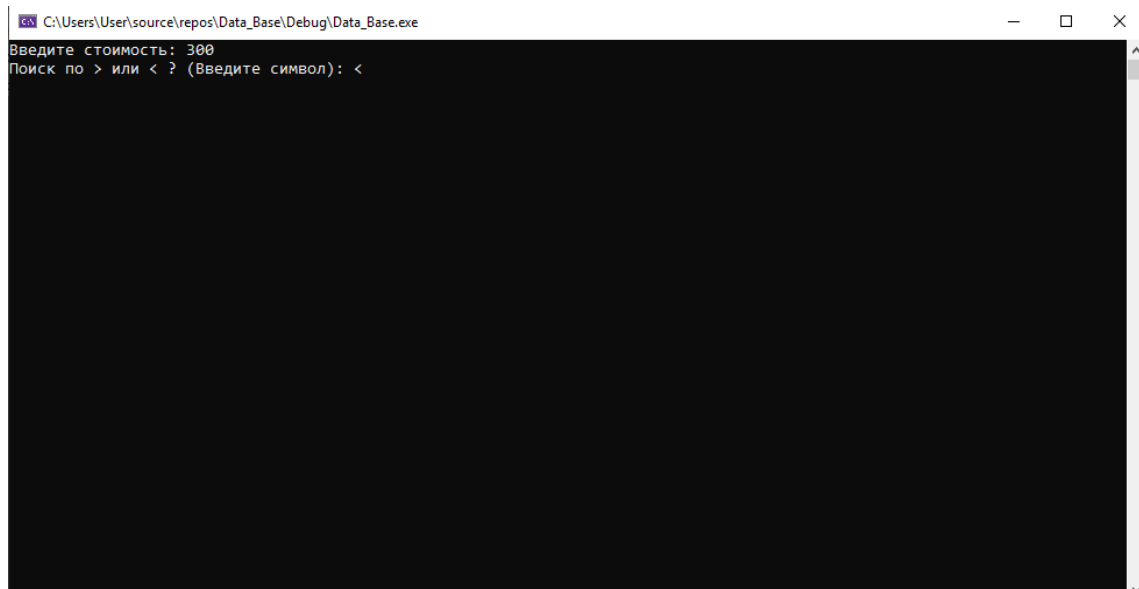


Рисунок - Сортировка по стоимости. Знак "<"

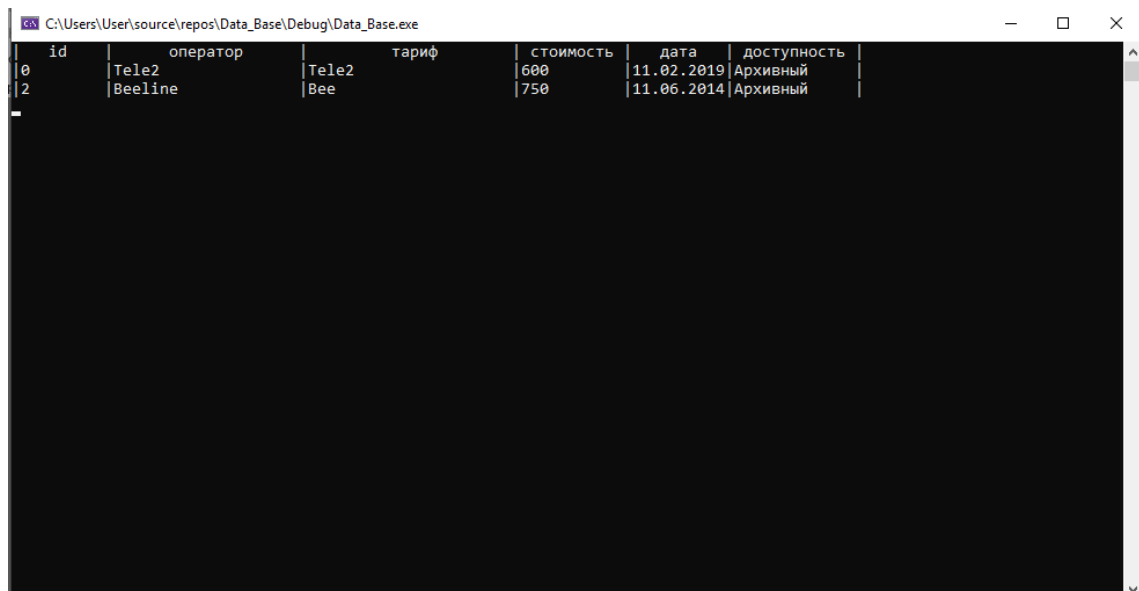
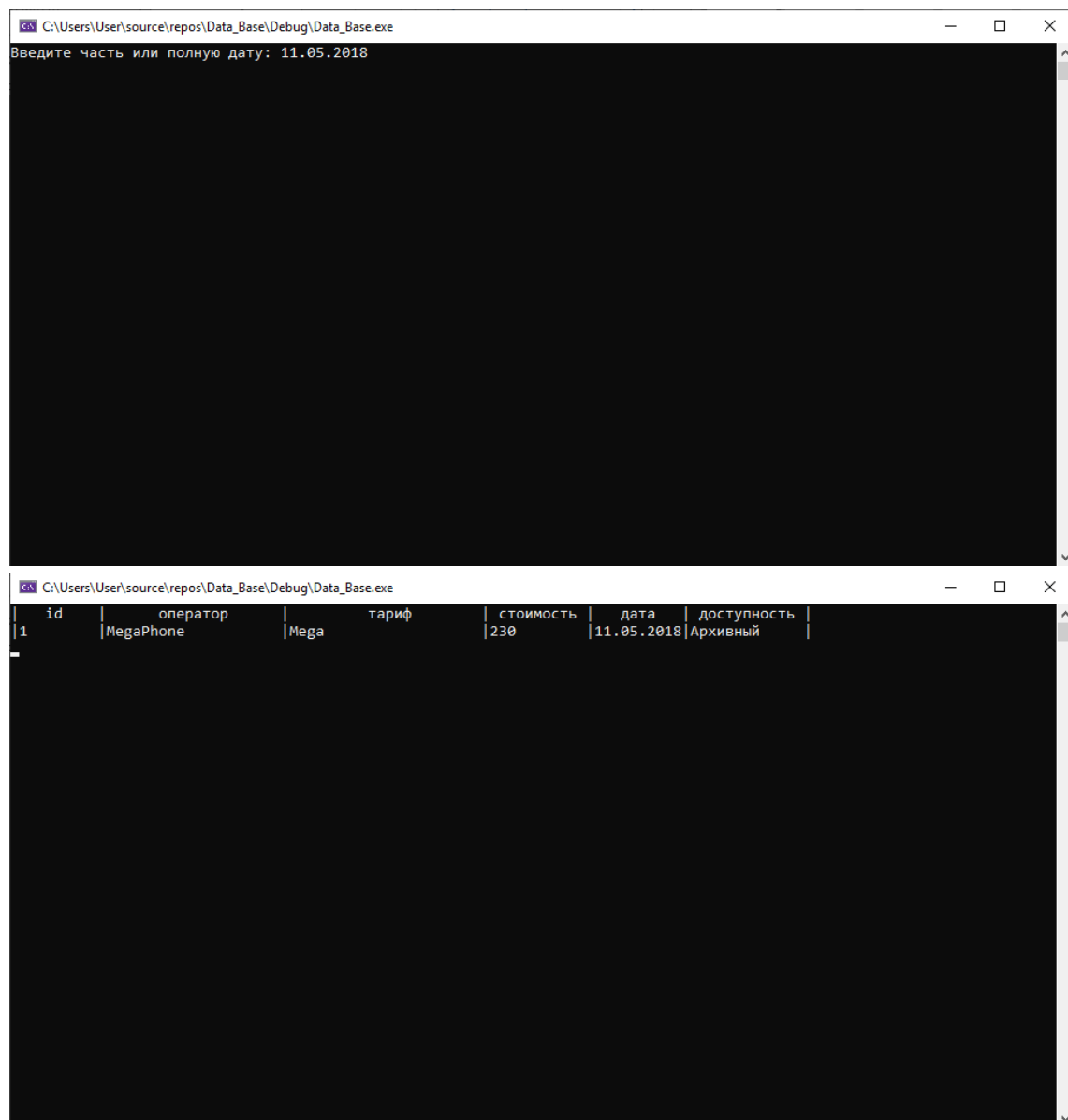


Рисунок - Результат сортировки по стоимости. Знак "<"

Поиск по дате предназначен для поиска группы записей по конкретной дате. Пользователю необходимо ввести дату и нажать клавишу Enter. Если записи с данным тарифом есть в базе данных, то на экран выведется отфильтрованный список с записями, в ином случае программа выдаст ошибку «База данных пуста».



Поиск по доступности предназначен для поиска доступных тарифных планов сотовых операторов. Пользователю необходимо ввести букву «Д», «д», «У», «у» для нахождения доступных тарифов или «Н», «н», «N», «n» для недоступных и нажать клавишу Enter. Если записи с данным тарифом есть в базе данных, то на экран выведется отфильтрованный список с записями, в ином случае программа выдаст ошибку «База данных пуста».

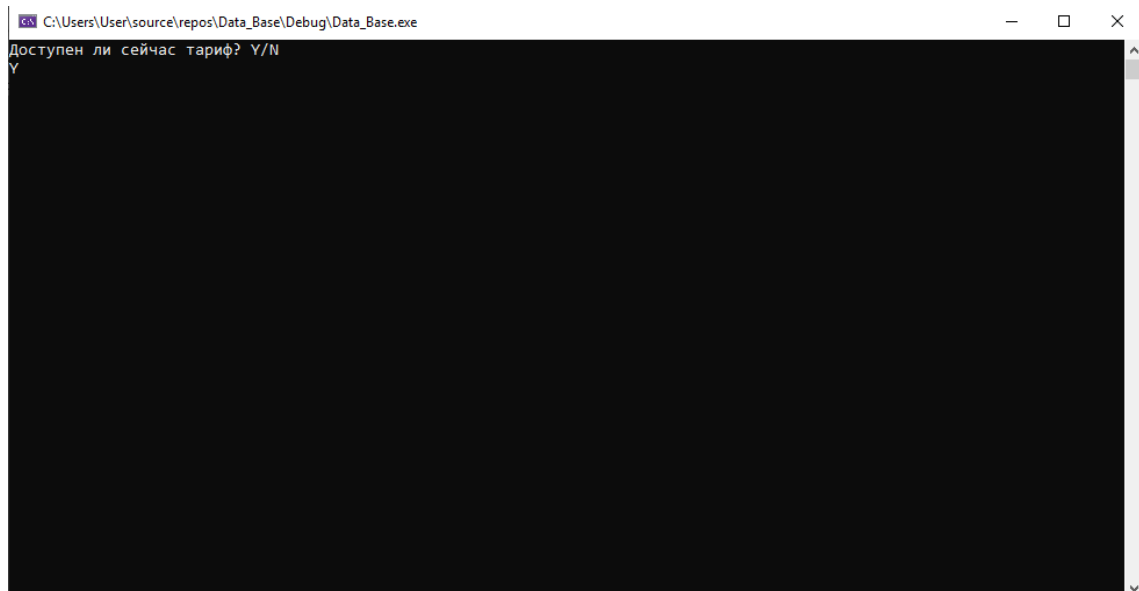


Рисунок - Поиск доступных записей

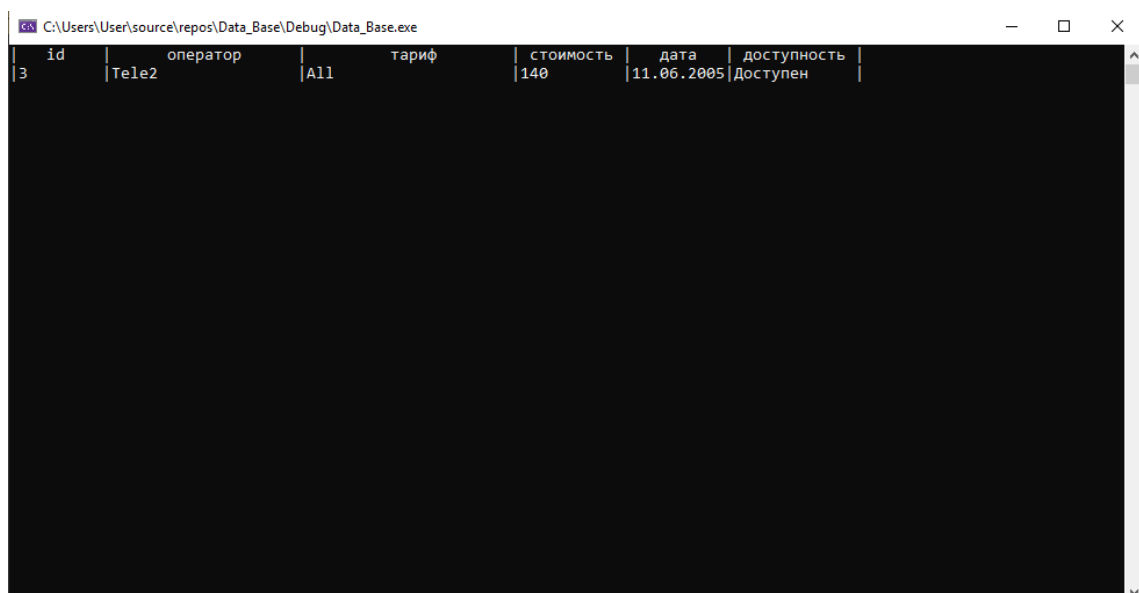


Рисунок - Результат поиска доступных записей

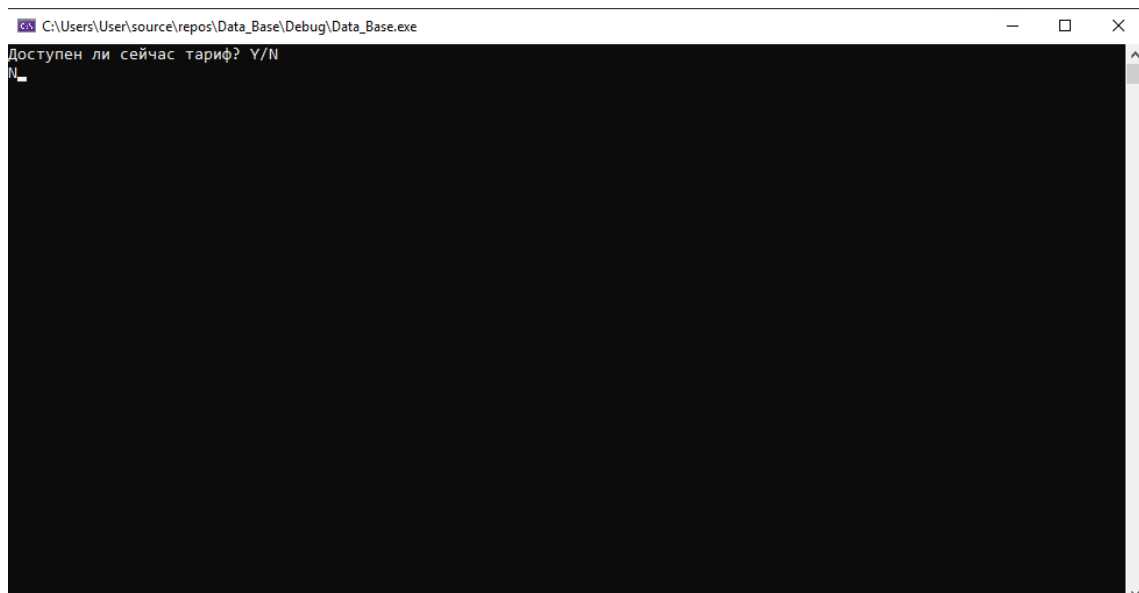


Рисунок - Поиск архивных записей

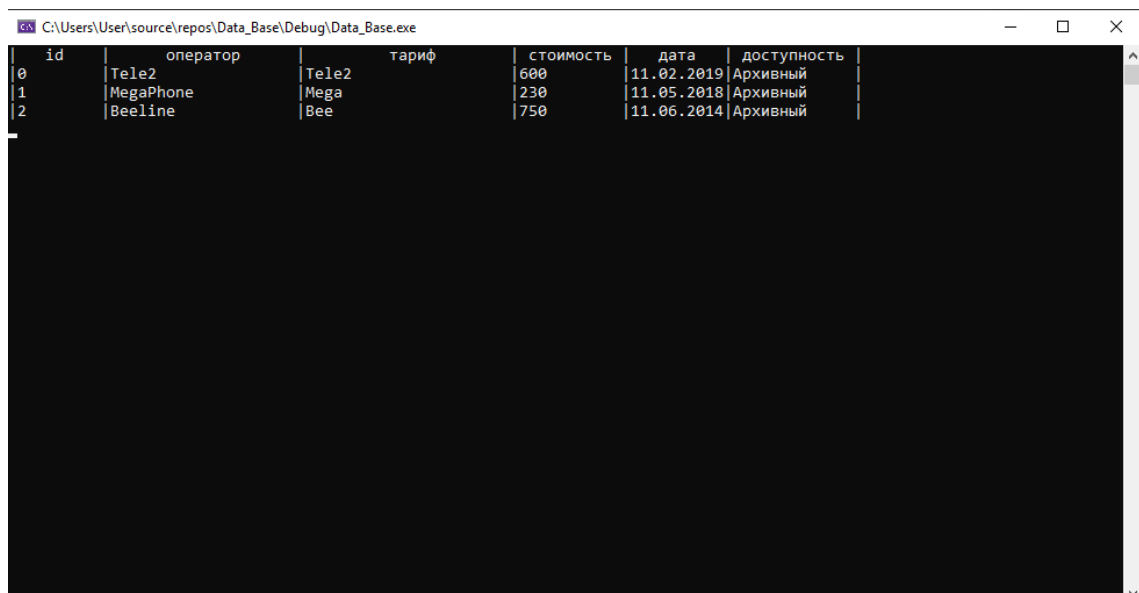


Рисунок - Результат поиска архивных записей

Для выхода из программы необходимо выбрать пункт меню «Выход» и нажать клавишу Enter.

Заключение

При выполнении данной курсовой работы были получены навыки разработки многомодульных программ. Была освоена работа с файлами, изучены функции работы с консолью и оперативной памятью. Также были получены основные навыки отладки и тестирования программ и программирования в среде VisualStudio 2019 на языках Си/C++ и Assembler.

В рамках курсовой работы была написана программа для работы с базой данных сотовых операторов. Она имеет все необходимые функции для работы с базами данных.

В дальнейшем программу можно улучшить, путём добавления графического интерфейса, что позволит улучшить навигацию по меню программы, путём добавления возможности использования мыши.

Список используемых источников

1. Керниган Б. Ритчи Д. Язык программирования С. 1985 г.
2. MSDN.
3. Прата С. Язык программирования C++. 2019 г.
4. А.А. Тюгашев. Языки программирования. Учебное пособие. 2018 г.
5. Руслан Аблязов. Программирование на ассемблере на платформе x86-64. 2016 г.

Приложение А

Листинги программы

Файл Data_Base.cpp

```
#include<locale.h>
#include<windows.h>

//Подключение собственных модулей
#include"color_print.h"

#include"init.h"
#include"append.h"

#include"change_example.h"
#include"add_provider.h"
#include"print_database.h"
#include"del_example.h"

#include"write_to_file.h"
#include"read_from_file.h"

#include"consts.h"

short ERROR_LOG[50];
short url_error = 0;

intmain()
{
    // Настройка поддержки русского языка
    setlocale(LC_ALL, "Rus");
    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);

    Provider* Head = NULL;
    char number_of_answer;
    bool Fbreak = 0;

    //Титульник.
    printf("Курсовая          работа\nПо          дисциплине\n\"Программирование\"\n\nНа          тему:          \"База          данных          сотовых\nоператоров\"\n\n\nВыполнил          студент          группы          18BB2:          Лехов\nКирилл\nПринял: Федюнин Р.Н.");
    GETCHAR_HANDLER = getchar();
    system("cls");

    while (1)
    {
        // Вывод меню
        printf("Выберите нужный вариант:\n");
        printf("1) Добавить запись\n");
```

```

printf("2) Вывести записи\n");
printf("3) Удалить запись\n");
printf("4) Изменить запись\n");
printf("5) Загрузить БД в файл\n");
printf("6) Загрузить БД из файла\n");
printf("0) Выход\n");

scanf_s("%c", &number_of_answer, 1);
GETCHAR_HANDLER = getchar();

switch (number_of_answer)
{
case ('1'):
{
    try
    {
        Head = add_provider(Head);
        color_print(SUCCESS, 2);
    }
    catch (short error)
    {
        color_print(FAIL, 3);
        ERROR_LOG[url_error] = error;
        url_error++;
    }

    GETCHAR_HANDLER = getchar();
    break;
}
case ('2'):
{
    print_database(Head);
    break;
}
case ('3'):
{
    Head = del_example(Head);
    break;
}
case ('4'):
{
    change_example(Head);
    break;
}
case ('5'):
{
    write_to_file(Head);
    break;
}
case ('6'):
{
    Head = read_from_file(Head);
    break;
}
}

```

```

    }
    case ('0'):
    {
        Fbreak = 1;
        break;
    }
    case ('9'):
    {
        __asm
        {
            push 1700;
            push 1500;
            call Beep

            push 500;
            push 5000;
            call Beep
        }
    }
    default:
        break;
}

system("cls");
if (Fbreak)
    break;
}
}

```

Файлconstс.cpp

```

#include "consts.h"

charSUCCESS[] = "Операция выполнена успешно!";
charFAIL[] = "При выполнении операции произошла ошибка.";
charNUL[] = "База данных пуста";
charFALSE_ID[] = "Данный ID не найден";
constchar* HEAD_OF_TABLE = "| id | оператор |
тариф | стоимость | дата | доступность |\n";
const char* FORMAT_OF_TABLE = "|%-9d|%-20s|%-22s|%-11d|%-
10s|%-13s|\n";

char GETCHAR_HANDLER;

```

Файлadd_provider.cpp

```

#include <stdio.h>
#include <windows.h>

#include "consts.h"
#include "crop_string.h"
#include "answer_handler.h"

#include "init.h"
#include "append.h"

Provider* add_provider(Provider* head_url)

```

```

{
    char telecom_operator[LEN_STRING];
    char tariff[LEN_STRING];
    int price;
    char data[LEN_DATA];
    char* url_data = data;
    bool available = 1;

    char answer;
    bool fbreak = 0;

    system("cls");
    printf("Введите оператора: ");
    fgets(telecom_operator, LEN_STRING, stdin);
    crop(telecom_operator);

    printf("Введите тариф: ");
    fgets(tariff, LEN_STRING, stdin);
    crop(tariff);

    printf("Введите стоимость в рублях: ");
    scanf_s("%d", &price);
    GETCHAR_HANDLER = getchar();

    printf("Введите дату в формате DD.MM.YY\n");
    fgets(data, LEN_DATA, stdin);

    while (1)
    {
        printf("Доступен ли сейчас тариф? Y/N\n");
        answer = answer_handler();
        switch (answer)
        {
            case (0):
                available = 0;
                fbreak = 1;
                break;
            case (1):
                available = 1;
                fbreak = 1;
                break;
            case (-1):
                GETCHAR_HANDLER = getchar();
                printf("Повторите попытку. \nДоступен ли сейчас
тариф? Y/N\n");
                };

            if (fbreak)
                break;
        }

        if (head_url == NULL)

```

```

        head_url = init(0, telecom_operator, tariff, price,
data, available);
    else
    {
        Provider* head = head_url;
        do
        {
            if (head_url->next == NULL)
            {
                append(head_url->id + 1, telecom_operator,
tariff, price, data, available, head_url);
                break;
            }

            if (head_url->next->id == -1)
            {
                head_url->next->id = head_url->id + 1;
                strcpy_s(head_url->next->telecom_operator,
LEN_STRING, telecom_operator);
                strcpy_s(head_url->next->tariff,
LEN_STRING, tariff);
                head_url->next->price = price;
                strcpy_s(head_url->next->data, LEN_DATA,
data);
                head_url->next->available = available;
                break;
            }

            head_url = head_url->next;
        } while (head_url != NULL);
        head_url = head;
    }

    GETCHAR_HANDLER = getchar();
    system("cls");
    return head_url;
}

```

Файл init.cpp

```

#include <string.h>
#include <stdlib.h>

#include "consts.h"

Provider* init(int id, char* telecom_operator, char* tariff,
int price, char* data, bool available)
{
    Provider* url = (Provider*)malloc(sizeof(Provider));

    if (url)
    {
        url->id = id;
        strcpy_s(url->telecom_operator, LEN_STRING,
telecom_operator);
    }
}

```



```

        strcpy_s(url->tariff, LEN_STRING, tariff);
        url->price = price;
        strcpy_s(url->data, LEN_DATA, data);
        url->available = available;
        url->next = NULL;

        return url;
    }
    else
    {
        return NULL;
    }
}

```

Файл append.cpp

```

#include <stdio.h>
#include <string.h>
#include <stdlib.h>

#include "consts.h"

void append(int id, char* telecom_operator, char* tariff, int
price, char* data, bool available, Provider* head_url)
{
    Provider* url = (Provider*)malloc(sizeof(Provider));

    if (url)
    {
        url->id = id;
        strcpy_s(url->telecom_operator,          LEN_STRING,
telecom_operator);
        strcpy_s(url->tariff, LEN_STRING, tariff);
        url->price = price;
        strcpy_s(url->data, LEN_DATA, data);
        url->available = available;
        url->next = NULL;

        Provider* last_url = head_url;
        while (last_url->next != NULL)
            last_url = last_url->next;

        last_url->next = url;
    }
}

```

Файл change_example.cpp

```

#include <stdio.h>
#include <windows.h>
#include <string.h>

#include "color_print.h"
#include "consts.h"
#include "crop_string.h"
#include "answer_handler.h"

```

```

#include "change_example.h"

void change_example(Provider* head_url)
{
    system("cls");

    if (head_url == NULL)
    {
        color_print(NUL, 1);
        GETCHAR_HANDLER = getchar();

        return;
    }

    char telecom_operator[LEN_STRING];
    char tariff[LEN_STRING];
    int price;
    char data[LEN_DATA];
    char* url_data = data;
    bool available = 1;

    char answer;
    bool fbreak = 0;
    int id;
    Provider* changer = NULL;

    printf("Введите id изменяемого экземпляра: ");
    scanf_s("%d", &id);
    GETCHAR_HANDLER = getchar();

    do
    {
        if (head_url->id == id)
        {
            changer = head_url;
        }

        head_url = head_url->next;
    } while (head_url != NULL);

    if (id == -1)
        changer = NULL;

    if (changer == NULL)
    {
        system("cls");
        color_print(FALSE_ID, 1);
        GETCHAR_HANDLER = getchar();
        return;
    }

    printf("Введите оператора: ");
    fgets(telecom_operator, LEN_STRING, stdin);

```

```

crop(telecom_operator);

printf("Введите тариф: ");
fgets(tariff, LEN_STRING, stdin);
crop(tariff);

printf("Введите стоимость в рублях: ");
scanf_s("%d", &price);
GETCHAR_HANDLER = getchar();

printf("Введите дату в формате DD.MM.YY\n");
fgets(data, LEN_DATA, stdin);

while (1)
{
    printf("Доступен ли сейчас тариф? Y/N\n");
    answer = answer_handler();
    switch (answer)
    {
        case (0):
            available = 0;
            fbreak = 1;
            break;
        case (1):
            available = 1;
            fbreak = 1;
            break;
        case (-1):
            GETCHAR_HANDLER = getchar();
            printf("Повторите попытку. \nДоступен ли сейчас
тариф? Y/N\n");
    };

    if (fbreak)
        break;
}

strcpy_s(changer->telecom_operator, LEN_STRING,
telecom_operator);
strcpy_s(changer->tariff, LEN_STRING, tariff);
strcpy_s(changer->data, LEN_DATA, data);

changer->price = price;
changer->available = available;

system("cls");
color_print(SUCCESS, 2);
GETCHAR_HANDLER = getchar();
GETCHAR_HANDLER = getchar();
return;
}

```

Файл del_example.cpp

```

#include <stdio.h>
#include <windows.h>

#include "consts.h"
#include "color_print.h"

Provider* del_example(Provider* head_url)
{
    system("cls");
    if (head_url == NULL)
    {
        color_print(NUL, 1);
        GETCHAR_HANDLER = getchar();

        return NULL;
    }

    int id;
    Provider* last_url = NULL;
    Provider* first_head_url = head_url;

    printf("Введите ID:");
    scanf_s("%d", &id);
    GETCHAR_HANDLER = getchar();

    do
    {
        if (head_url->id == id)
        {
            last_url = head_url;
        }

        head_url = head_url->next;
    } while (head_url != NULL);

    if (id == -1)
        last_url = NULL;

    if (last_url == NULL)
    {
        system("cls");
        color_print(FALSE_ID, 1);
        GETCHAR_HANDLER = getchar();
        return first_head_url;
    }

    head_url = first_head_url;
    last_url = NULL;

    if (head_url->id == id)
        if (head_url->next == NULL)

```

```

        {
            free(head_url);
            system("cls");
            color_print(SUCCESS, 2);
            GETCHAR_HANDLER = getchar();
            return NULL;
        }
        else
        {
            head_url = head_url->next;
            system("cls");
            color_print(SUCCESS, 2);
            GETCHAR_HANDLER = getchar();
            return head_url;
        }
    }

    do
    {
        if (head_url->id == id)
        {
            head_url->id = -1;

            system("cls");
            color_print(SUCCESS, 2);
            GETCHAR_HANDLER = getchar();
            return first_head_url;
        }

        last_url = head_url;
        head_url = head_url->next;
    } while (head_url != NULL);

    return first_head_url;
}

```

Файл print_database.cpp

```

#include <stdio.h>
#include <windows.h>
#include <stdlib.h>

#include "consts.h"
#include "color_print.h"
#include "answer_handler.h"

int compare_id(const void* x1, const void* x2)
{
    return ((const struct Provider*)x2)->id - ((const struct
Provider*)x1)->id;
}

void bubbleSort(Provider* temp[], int size)
{

```

```

    long i, j;
    Provider* x, a, b;

    for (i = 0; i < size; i++)
    {
        for (j = size - 1; j > i; j--)
        {
            a = *temp[j - 1];
            b = *temp[j];
            if (a.price > b.price)
            {
                x = temp[j - 1];
                temp[j - 1] = temp[j];
                temp[j] = x;
            }
        }
    }
}

void bubbleSort_back(Provider* temp[], int size)
{
    long i, j;
    Provider* x;
    Provider a, b;

    for (i = 0; i < size; i++)
    {
        for (j = size - 1; j > i; j--)
        {
            b = *temp[j - 1];
            a = *temp[j];
            if (a.price > b.price)
            {
                x = temp[j - 1];
                temp[j - 1] = temp[j];
                temp[j] = x;
            }
        }
    }
}

void print_database(Provider* head_url)
{
    system("cls");

    if (head_url == NULL)
    {
        color_print(NUL, 1);
        GETCHAR_HANDLER = getchar();

        return;
    }
}

```

```

    }

    char avail[9];
    char answer;
    char answer_2;

    int id;
    Provider* checker = NULL;

    //Меню фильтров.
    printf("1) Показать все записи\n");
    printf("2) Поиск по фильтру\n");
    printf("3) Отсортировать записи\n");
    printf("0) Выход\n");

    scanf_s("%c", &answer, 1);
    GETCHAR_HANDLER = getchar();
    system("cls");

    switch (answer)
    {
    case ('1'):
    {
        printf(HEAD_OF_TABLE);

        do
        {
            if (head_url->id == -1)
            {
                head_url = head_url->next;
                continue;
            }

            if (head_url->available)
            {
                strcpy_s(avail, 9, "Доступен");
            }
            else
            {
                strcpy_s(avail, 9, "Архивный");
            }

            printf(FORMAT_OF_TABLE, head_url->id, head_url->telecom_operator, head_url->tariff, head_url->price, head_url->data, avail);

            head_url = head_url->next;
        } while (head_url != NULL);
        GETCHAR_HANDLER = getchar();
        return;
    }
    case ('2'):
    {
        system("cls");
    }
    }

```

```

printf("1) Поиск по ID\n");
printf("2) Поиск по оператору\n");
printf("3) Поиск по тарифу\n");
printf("4) Поиск по стоимости\n");
printf("5) Поиск по дате\n");
printf("6) Поиск по доступности\n");
printf("0) Выход\n");
//Меню фильтров.
scanf_s("%c", &answer_2, 1);
GETCHAR_HANDLER = getchar();
system("cls");

switch (answer_2)
{
case ('1'):
{
    printf("Введите ID:");
    scanf_s("%d", &id);
    GETCHAR_HANDLER = getchar();

    do
    {
        if (head_url->id == id)
        {
            checker = head_url;
        }

        head_url = head_url->next;
    } while (head_url != NULL);

    if (id == -1)
        checker = NULL;

    if (checker == NULL)
    {
        system("cls");
        color_print(FALSE_ID, 1);
        GETCHAR_HANDLER = getchar();
        return;
    }

    system("cls");
    printf(HEAD_OF_TABLE);

    if (checker->available)
    {
        strcpy_s(avail, 9, "Доступен");
    }
    else
    {
        strcpy_s(avail, 9, "Архивный");
    }
}
}

```



```

        printf(FORMAT_OF_TABLE, checker->id, checker-
>telecom_operator, checker->tariff, checker->price, checker->data,
avail);

        GETCHAR_HANDLER = getchar();
        break;
    }
    case ('2'):
    {
        char string[LEN_STRING];
        Provider* head = head_url;
        int number_of_examples = 0;
        int number_of_string = 0;
        boolflag;

        system("cls");
        printf("Введите часть или полное название: ");
        scanf_s("%s", string, LEN_STRING);
        system("cls");
        GETCHAR_HANDLER = getchar();

        //поиск по фильтрам
        do
        {
            number_of_string = 0;
            flag = 0;

            if (head->id == -1)
            {
                head = head->next;
                continue;
            }

            do
            {
                if (string[number_of_string] == head-
>telecom_operator[number_of_string])
                {
                    flag = 1;
                    number_of_string++;

                    if (string[number_of_string] ==
'\0')
                        break;
                }
                else
                {
                    if ((string[number_of_string] ==
'\0') & flag)
                    {
                        flag = 1;
                        break;
                    }
                }
            }
        }
    }

```

```

                                if ((head-
>telecom_operator[number_of_string] == '\0') & flag)
                                {
                                    flag = 1;
                                    break;
                                }

                                flag = 0;
                                break;
                            }

                        } while (1);

                    if (flag)
                    {
                        number_of_examples++;
                    }

                    head = head->next;
                } while (head != NULL);

                if (number_of_examples == 0)
                {
                    color_print(NUL, 1);
                    GETCHAR_HANDLER = getchar();
                    return;
                }

                head = head_url;
                Provider** heads = new Provider*
[number_of_examples];
                number_of_examples = 0;

                do
                {
                    number_of_string = 0;
                    flag = 0;

                    if (head->id == -1)
                    {
                        head = head->next;
                        continue;
                    }

                    do
                    {
                        if (string[number_of_string] == head-
>telecom_operator[number_of_string])
                        {
                            flag = 1;
                            number_of_string++;

```

```

                                if (string[number_of_string] ==
'\0')
                                break;
                                }
                                else
                                {
                                if ((string[number_of_string] ==
'\0') & flag)
                                {
                                    flag = 1;
                                    break;
                                }

                                if ((head->telecom_operator[number_of_string] == '\0') & flag)
                                {
                                    flag = 1;
                                    break;
                                }

                                flag = 0;
                                break;
                                }

                                } while (1);

                                if (flag)
                                {
                                    heads[number_of_examples] = head;
                                    number_of_examples++;
                                }

                                head = head->next;

                                } while (head != NULL);

                                printf(HEAD_OF_TABLE);
                                for (int i = 0; i < number_of_examples; i++)
                                {
                                    if (heads[i]->available)
                                    {
                                        strcpy_s(avail, 9, "Доступен");
                                    }
                                    else
                                    {
                                        strcpy_s(avail, 9, "Архивный");
                                    }

                                    printf(FORMAT_OF_TABLE, heads[i]->id,
heads[i]->telecom_operator, heads[i]->tariff, heads[i]->price,
heads[i]->data, avail);
                                }
                                GETCHAR_HANDLER = getchar();

```

```

        break;
    }
    case ('3'):
    {
        char string[LEN_STRING];
        Provider* head = head_url;
        int number_of_examples = 0;
        int number_of_string = 0;
        boolflag;

        system("cls");
        printf("Введите часть или полное название: ");
        scanf_s("%s", string, LEN_STRING);
        system("cls");
        GETCHAR_HANDLER = getchar();

        //поиск по фильтрам
        do
        {
            number_of_string = 0;
            flag = 0;

            if (head->id == -1)
            {
                head = head->next;
                continue;
            }

            do
            {
                if (string[number_of_string] == head-
>tariff[number_of_string])
                {
                    flag = 1;
                    number_of_string++;

                    if (string[number_of_string] ==
'\0')
                        break;
                }
                else
                {
                    if ((string[number_of_string] ==
'\0') & flag)

                    {
                        flag = 1;
                        break;
                    }

                    if ((head-
>tariff[number_of_string] == '\0') & flag)
                    {
                        flag = 1;

```

```

        break;
    }

    flag = 0;
    break;
}

} while (1);

if (flag)
{
    number_of_examples++;
}

head = head->next;
} while (head != NULL);

if (number_of_examples == 0)
{
    color_print(NUL, 1);
    GETCHAR_HANDLER = getchar();
    return;
}

head = head_url;
Provider** heads = new Provider *
[number_of_examples];
number_of_examples = 0;

do
{
    number_of_string = 0;
    flag = 0;

    if (head->id == -1)
    {
        head = head->next;
        continue;
    }

    do
    {
        if (string[number_of_string] == head-
>tariff[number_of_string])
        {
            flag = 1;
            number_of_string++;

            if (string[number_of_string] ==
'\0')
                break;
        }
        else

```

```

        {
            if ((string[number_of_string] ==
'\0') & flag)
            {
                flag = 1;
                break;
            }

            if ((head-
>tariff[number_of_string] == '\0') & flag)
            {
                flag = 1;
                break;
            }

            flag = 0;
            break;
        }

    } while (1);

    if (flag)
    {
        heads[number_of_examples] = head;
        number_of_examples++;
    }

    head = head->next;

} while (head != NULL);

printf(HEAD_OF_TABLE);
for (int i = 0; i < number_of_examples; i++)
{
    if (heads[i]->available)
    {
        strcpy_s(avail, 9, "Доступен");
    }
    else
    {
        strcpy_s(avail, 9, "Архивный");
    }

    printf(FORMAT_OF_TABLE, heads[i]->id,
heads[i]->telecom_operator, heads[i]->tariff, heads[i]->price,
heads[i]->data, avail);
}
GETCHAR_HANDLER = getchar();
break;
}
case ('4'):
{
    Provider* head = head_url;

```

```

int number_of_examples = 0;
bool fbreak = 0;
int price = 0;
char sign;

system("cls");
printf("Введите стоимость: ");
scanf_s("%d", &price);
GETCHAR_HANDLER = getchar();
printf("Поискпо>или< ? (Введите символ): ");
scanf_s("%c", &sign, 2);
GETCHAR_HANDLER = getchar();
system("cls");

do
{
    if (head->id == -1)
    {
        head = head->next;
        continue;
    }

    if (sign == '>')
    {
        if (head->price < price)
        {
            number_of_examples++;
        }
    }

    if (sign == '<')
    {
        if (head->price > price)
        {
            number_of_examples++;
        }
    }

    head = head->next;
} while (head != NULL);

if (number_of_examples == 0)
{
    color_print(NUL, 1);
    GETCHAR_HANDLER = getchar();
    return;
}

head = head_url;
Provider** heads = new Provider *
[number_of_examples];
number_of_examples = 0;

```

```

do
{
    if (head->id == -1)
    {
        head = head->next;
        continue;
    }

    if (sign == '>')
    {
        if (head->price < price)
        {
            heads[number_of_examples] =
head;
            number_of_examples++;
        }
    }

    if (sign == '<')
    {
        if (head->price > price)
        {
            heads[number_of_examples] =
head;
            number_of_examples++;
        }
    }

    head = head->next;
} while (head != NULL);

printf(HEAD_OF_TABLE);
for (int i = 0; i < number_of_examples; i++)
{
    if (heads[i]->available)
    {
        strcpy_s(avail, 9, "Доступен");
    }
    else
    {
        strcpy_s(avail, 9, "Архивный");
    }

    printf(FORMAT_OF_TABLE, heads[i]->id,
heads[i]->telecom_operator, heads[i]->tariff, heads[i]->price,
heads[i]->data, avail);
}
GETCHAR_HANDLER = getchar();
break;
}
case ('5'):

```



```

{
    char string[LEN_DATA];
    Provider* head = head_url;
    int number_of_examples = 0;
    int number_of_string = 0;
    bool flag;

    system("cls");
    printf("Введите часть или полную дату: ");
    scanf_s("%s", string, LEN_DATA);
    system("cls");
    GETCHAR_HANDLER = getchar();

    //поиск по фильтрам
    do
    {
        number_of_string = 0;
        flag = 0;

        if (head->id == -1)
        {
            head = head->next;
            continue;
        }

        do
        {
            if (string[number_of_string] == head-
>data[number_of_string])
            {
                flag = 1;
                number_of_string++;

                if (string[number_of_string] ==
'\0')
                    break;
            }
            else
            {
                if ((string[number_of_string] ==
'\0') & flag)
                {
                    flag = 1;
                    break;
                }

                if ((head-
>data[number_of_string] == '\0') & flag)
                {
                    flag = 1;
                    break;
                }
            }
        }
    }
}

```

```

        flag = 0;
        break;
    }

    } while (1);

    if (flag)
    {
        number_of_examples++;
    }

    head = head->next;
} while (head != NULL);

if (number_of_examples == 0)
{
    color_print(NUL, 1);
    GETCHAR_HANDLER = getchar();
    return;
}

head = head_url;
Provider** heads = new Provider *
[number_of_examples];
number_of_examples = 0;

do
{
    number_of_string = 0;
    flag = 0;

    if (head->id == -1)
    {
        head = head->next;
        continue;
    }

    do
    {
        if (string[number_of_string] == head-
>data[number_of_string])
        {
            flag = 1;
            number_of_string++;

            if (string[number_of_string] ==
'\0')
                break;
        }
        else
        {
            if ((string[number_of_string] ==
'\0') & flag)

```

```

        {
            flag = 1;
            break;
        }

        if ((head->data[number_of_string] == '\0') & flag)
        {
            flag = 1;
            break;
        }

        flag = 0;
        break;
    }

} while (1);

if (flag)
{
    heads[number_of_examples] = head;
    number_of_examples++;
}

head = head->next;

} while (head != NULL);

printf(HEAD_OF_TABLE);
for (int i = 0; i < number_of_examples; i++)
{
    if (heads[i]->available)
    {
        strcpy_s(avail, 9, "Доступен");
    }
    else
    {
        strcpy_s(avail, 9, "Архивный");
    }

    printf(FORMAT_OF_TABLE, heads[i]->id,
heads[i]->telecom_operator, heads[i]->tariff, heads[i]->price,
heads[i]->data, avail);
}
GETCHAR_HANDLER = getchar();
break;
}
case ('6'):
{
    Provider* head = head_url;
    int number_of_examples = 0;
    bool fbreak = 0;
    bool available;

```

```

system("cls");

while (1)
{
    printf("Доступен ли сейчас тариф? Y/N\n");
    answer = answer_handler();
    switch (answer)
    {
        case (0):
            available = 0;
            fbreak = 1;
            break;
        case (1):
            available = 1;
            fbreak = 1;
            break;
        case (-1):
            GETCHAR_HANDLER = getchar();
            system("cls");
            printf("Повторите попытку. \nДоступен
ли сейчас тариф? Y/N\n");
    };

    if (fbreak)
        break;
}
system("cls");
GETCHAR_HANDLER = getchar();

do
{
    if (head->id == -1)
    {
        head = head->next;
        continue;
    }

    if (available == head->available)
    {
        number_of_examples++;
    }

    head = head->next;
} while (head != NULL);

if (number_of_examples == 0)
{
    color_print(NUL, 1);
    GETCHAR_HANDLER = getchar();
    return;
}

```

```

        head = head_url;
        Provider** heads = new Provider *
[number_of_examples];
        number_of_examples = 0;

        do
        {
            if (head->id == -1)
            {
                head = head->next;
                continue;
            }

            if (available == head->available)
            {
                heads[number_of_examples] = head;
                number_of_examples++;
            }
            head = head->next;

        } while (head != NULL);

        printf(HEAD_OF_TABLE);
        for (int i = 0; i < number_of_examples; i++)
        {
            if (heads[i]->available)
            {
                strcpy_s(avail, 9, "Доступен");
            }
            else
            {
                strcpy_s(avail, 9, "Архивный");
            }

            printf(FORMAT_OF_TABLE, heads[i]->id,
heads[i]->telecom_operator, heads[i]->tariff, heads[i]->price,
heads[i]->data, avail);
        }
        GETCHAR_HANDLER = getchar();
        break;
    }
    case ('0'):
        break;
    };
    break;
}
case ('3'):
{
    // Вывод меню
    printf("1) Сортировать в обратном порядке по ID\n");
    printf("2) Сортировать по цене\n");

```

```

printf("3) Сортировать по цене в обратном
порядке\n");

scanf_s("%c", &answer_2, 1);
GETCHAR_HANDLER = getchar();
system("cls");

switch (answer_2)
{
case ('1'):
{
    int number_of_examples = 0;
    Provider* head = head_url;

    do
    {
        if (head_url->id == -1)
        {
            head_url = head_url->next;
            continue;
        }

        head_url = head_url->next;
        number_of_examples++;
    } while (head_url != NULL);

    Provider** id_sort_back =
(Provider**)malloc(sizeof(Provider*) * number_of_examples);
    head_url = head;
    number_of_examples = 0;

    do
    {
        if (head_url->id == -1)
        {
            head_url = head_url->next;
            continue;
        }

        id_sort_back[number_of_examples] =
head_url;

        head_url = head_url->next;
        number_of_examples++;
    } while (head_url != NULL);

    qsort(id_sort_back, number_of_examples,
sizeof(Provider*), compare_id);

    printf(HEAD_OF_TABLE);
    for (int i = 0; i < number_of_examples; i++)
    {
        if (id_sort_back[i]->available)
        {

```

```

        strcpy_s(avail, 9, "Доступен");
    }
    else
    {
        strcpy_s(avail, 9, "Архивный");
    }

    printf(FORMAT_OF_TABLE, id_sort_back[i]-
>id, id_sort_back[i]->telecom_operator, id_sort_back[i]->tariff,
id_sort_back[i]->price, id_sort_back[i]->data, avail);
    }

    free(id_sort_back);
    GETCHAR_HANDLER = getchar();
    break;
}
case ('2'):
{
    int number_of_examples = 0;
    Provider* head = head_url;

    do
    {
        if (head_url->id == -1)
        {
            head_url = head_url->next;
            continue;
        }
        head_url = head_url->next;
        number_of_examples++;
    } while (head_url != NULL);

    Provider** price_sort = (Provider
**)malloc(sizeof(Provider*) * number_of_examples);
    head_url = head;
    number_of_examples = 0;

    do
    {
        if (head_url->id == -1)
        {
            head_url = head_url->next;
            continue;
        }

        price_sort[number_of_examples] = head_url;
        head_url = head_url->next;
        number_of_examples++;
    } while (head_url != NULL);

    bubbleSort(price_sort, number_of_examples);

    printf(HEAD_OF_TABLE);

```

```

        for (int i = 0; i < number_of_examples; i++)
        {
            if (price_sort[i]->available)
            {
                strcpy_s(avail, 9, "Доступен");
            }
            else
            {
                strcpy_s(avail, 9, "Архивный");
            }

            printf(FORMAT_OF_TABLE, price_sort[i]->id,
price_sort[i]->telecom_operator, price_sort[i]->tariff,
price_sort[i]->price, price_sort[i]->data, avail);
        }

        free(price_sort);
        GETCHAR_HANDLER = getchar();
        break;
    }
    case ('3'):
    {
        int number_of_examples = 0;
        Provider* head = head_url;

        do
        {
            if (head_url->id == -1)
            {
                head_url = head_url->next;
                continue;
            }

            head_url = head_url->next;
            number_of_examples++;
        } while (head_url != NULL);

        Provider** price_sort_back = (Provider *
*)malloc(sizeof(Provider*) * number_of_examples);
        head_url = head;
        number_of_examples = 0;

        do
        {
            if (head_url->id == -1)
            {
                head_url = head_url->next;
                continue;
            }

            price_sort_back[number_of_examples] =
head_url;
            head_url = head_url->next;

```



```

        number_of_examples++;
    } while (head_url != NULL);

    bubbleSort_back(price_sort_back,
number_of_examples);

    printf(HEAD_OF_TABLE);
    for (int i = 0; i < number_of_examples; i++)
    {
        if (price_sort_back[i]->available)
        {
            strcpy_s(avail, 9, "Доступен");
        }
        else
        {
            strcpy_s(avail, 9, "Архивный");
        }

        printf(FORMAT_OF_TABLE,
price_sort_back[i]->id, price_sort_back[i]->telecom_operator,
price_sort_back[i]->tariff, price_sort_back[i]->price,
price_sort_back[i]->data, avail);
    }

    free(price_sort_back);
    GETCHAR_HANDLER = getchar();
    break;
}
}
}
case ('0'):
    return;
}

}

```

Файл read_from_file.cpp

```

#include <stdio.h>
#include <windows.h>

#include "consts.h"
#include "color_print.h"
#include "init.h"
#include "append.h"

Provider* read_from_file(Provider* head_url)
{
    char file_name[20];
    char error_of_read_from_file[] = "Ошибка при чтении из
файла";
    char success_read_from_file[] = "База данных загружена";
    FILE* data;

    char telecom_operator[LEN_STRING];

```

```

char tariff[LEN_STRING];
int price;
char data_s[LEN_DATA];
int avail = 0;
bool available = 1;
char er;
bool fnew = 0;

Provider* head;

system("cls");
printf("Введите название файла: ");
do
{
    scanf_s("%s",
(unsigned)_countof(file_name), &file_name,
    if (file_name[19] == '\\0')
    {
        system("cls");
        color_print(error_of_read_from_file, 1);
        return head_url;
    }

    if (fopen_s(&data, file_name, "r") == 0)
    {
        break;
    }
    else
    {
        system("cls");
        printf("Файла с таким названием не существует.
\\nВведите название файла: ");
    }
} while (1);
fclose(data);

fopen_s(&data, file_name, "r");

head = head_url;

fscanf_s(data, "-ST@RT-\\n");

do
{
    available = 0;
    er = fscanf_s(data, "%s\\n", telecom_operator,
LEN_STRING);

    if (er == -1)
    {
        system("cls");
        if (head != NULL)
        {

```

```

        color_print(success_read_from_file, 0);
    }
    else
    {
        color_print(error_of_read_from_file, 3);
    }

    GETCHAR_HANDLER = getchar();
    GETCHAR_HANDLER = getchar();

    fclose(data);
    return head_url;
}

fscanf_s(data, "%s\n", tariff, LEN_STRING);
fscanf_s(data, "%d\n", &price);
fscanf_s(data, "%s\n", data_s, LEN_DATA);
fscanf_s(data, "%d\n", &avail, 2);
fscanf_s(data, "-CONTINUE-\n");

if (avail > 0)
    available = 1;

if (!fnew)
{
    head_url = init(0, telecom_operator, tariff,
price, data_s, available);
    head = head_url;
    fnew = 1;
}
else
{
    do
    {
        if (head->next == NULL)
        {
            append(head->id + 1,
telecom_operator, tariff, price, data_s, available, head_url);
            break;
        }

        head = head->next;
    } while (1);
}
} while (1);

return head_url;
}

```

Файл write_to_file.cpp

```

#include <stdio.h>
#include <windows.h>

```

```

#include "consts.h"
#include "color_print.h"

void write_to_file(Provider* head_url)
{
    system("cls");

    if (head_url == NULL)
    {
        color_print(FAIL, 1);
        GETCHAR_HANDLER = getchar();
        system("cls");
        return;
    }

    char file_name[20];
    FILE* data;
    char error_to_write_file[] = "Ошибка при записи в файл";
    bool file_here = 0;
    int avail = 0;

    printf("Введите название файла: ");
    scanf_s("%s", &file_name, (unsigned)_countof(file_name));
    if (file_name[19] == '\\0')
    {
        system("cls");
        color_print(error_to_write_file, 1);
        return;
    }

    fopen_s(&data, file_name, "r");

    if (data == 0)
        file_here = 0;
    else
        file_here = 1;

    fclose(data);

    if (file_here)
    {
        fopen_s(&data, file_name, "w");
    }
    else
    {
        fopen_s(&data, file_name, "a");
    }

    fprintf_s(data, "-ST@RT-\\n");

    do
    {
        if (head_url->id == -1)

```

```

        {
            head_url = head_url->next;
            continue;
        }

        if (head_url->available)
            avail = 1;

        fprintf_s(data, "%s\n%s\n%d\n%s\n%d\n-CONTINUE-\n",
head_url->telecom_operator, head_url->tariff, head_url->price,
head_url->data, avail);
        head_url = head_url->next;
        avail = 0;
    } while (head_url != NULL);
    fclose(data);

    system("cls");
    color_print(SUCCESS, 2);
    GETCHAR_HANDLER = getchar();
    GETCHAR_HANDLER = getchar();
    return;
}

```

Файл answer_handler.cpp

```

#include <stdio.h>

char answer_handler()
{
    char user_answer[3];

    scanf_s("%s", user_answer, 3);

    if ((user_answer[0] == 'Y') || (user_answer[0] == 'y') ||
(user_answer[0] == 'Д') || (user_answer[0] == 'д'))
    {
        return 1;
    }

    if ((user_answer[0] == 'N') || (user_answer[0] == 'n') ||
(user_answer[0] == 'H') || (user_answer[0] == 'h'))
    {
        return 0;
    }

    return -1;
}

```

Файл color_print.cpp

```

#include <windows.h>
#include <stdio.h>

void color_print(char* text, int color)
{
    HANDLE console = GetStdHandle(STD_OUTPUT_HANDLE);

```

```

switch (color) {
case 1:
    SetConsoleTextAttribute(console, FOREGROUND_RED);
    break;
case 2:
    SetConsoleTextAttribute(console, FOREGROUND_GREEN);
    break;
case 0:
    SetConsoleTextAttribute(console, FOREGROUND_RED |
FOREGROUND_GREEN | FOREGROUND_BLUE | FOREGROUND_INTENSITY);
    break;
default:
    SetConsoleTextAttribute(console, FOREGROUND_RED |
FOREGROUND_GREEN | FOREGROUND_BLUE | FOREGROUND_INTENSITY);
    break;
}

puts(text);
SetConsoleTextAttribute(console, FOREGROUND_RED |
FOREGROUND_GREEN | FOREGROUND_BLUE | FOREGROUND_INTENSITY);

return;
}

```

Файл crop_string.cpp

```

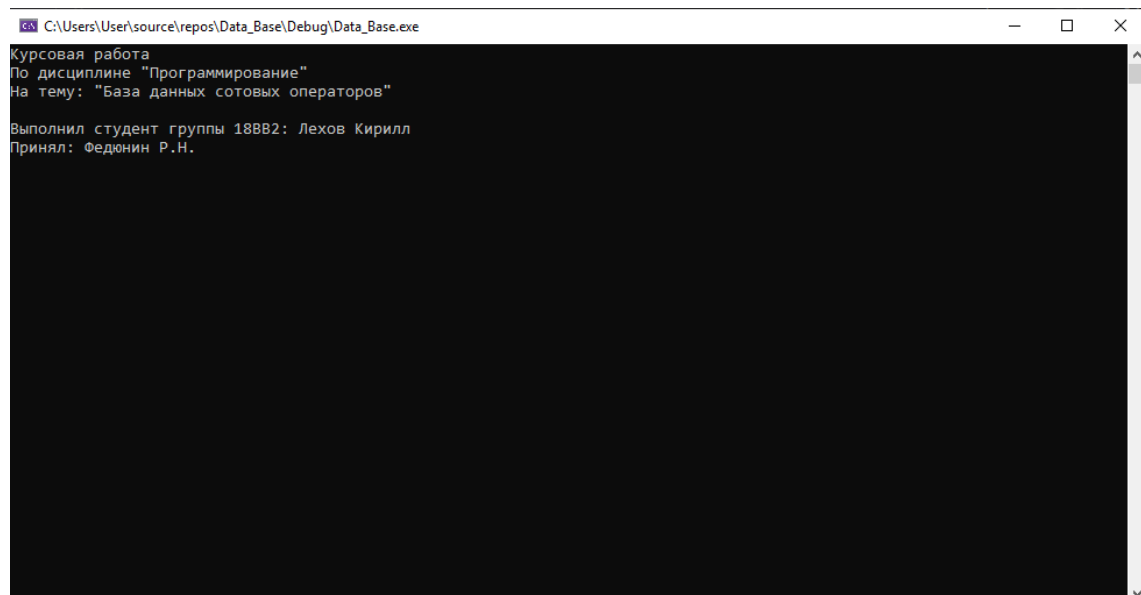
void crop(char* str)
{
    int str_num = 0;

    while (str[str_num] != '\0')
    {
        if (str[str_num] == char(10))
        {
            str[str_num] = char(0);
        };
        str_num++;
    };
}

```

Приложение В Результаты работы программы

Приложение В.1 – Информационная заставка



Приложение С

Результат работы программы

Файл test.bd

```
-ST@RT-  
Tele2  
Tele2  
600  
11.02.2019  
0  
-CONTINUE-  
MegaPhone  
Mega  
230  
11.05.2018  
0  
-CONTINUE-  
Beeline  
Bee  
750  
11.06.2014  
0  
-CONTINUE-  
Tele2  
All  
140  
11.06.2005  
1  
-CONTINUE-
```