## *КУРСОВА РOБОТА*

**КР.ІП –00.00.00.000 ПЗ**

**Група ІП-20-2**

**Балан Анастасія**

**2021**

Міністерство освіти і науки України

Івано-Франківський національний технічний університет нафти та газу

Інститут інформаційних технологій

**КУРСОВИЙ ПРОЄКТ**

з дисципліни «Об’єктно-орієнтоване програмування»

на тему: моделювання та аналіз АІС залізничної пасажирської станції

**Студента (ки) 1 курсу групи ІП-20-2**

**напряму підготовки** 121

інженерія програмного забезпечення

Балан А.І.

**(прізвище та ініціали)**

**Керівник\_**доцент Шекета В. І.

**(посада, вчене звання, науковий ступінь, прізвище та ініціали)**

**Національна шкала \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Кількість балів: \_\_\_ Оцінка: ECTS \_\_\_\_**

м. Івано-Франківськ — 2021рік

Міністерство освіти і науки України

Івано-Франківський національний технічний університет нафти та газу

Інститут інформаційних технологій

ЗАТВЕРДЖУЮ

зав. кафедри ІПЗ, проф., д.т.н.

**В.І. Шекета**

«\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2021р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

На курсовий проєкт з дисципліни «ООП» студентці Балан А.І. групи ІП-20-2.

ТЕМА: Розробка прикладної програми з проєктування інформаційних систем згідно завдання.

Постановка задачі.

1. Розробити прикладну програму
2. Розробити інтерфейс користувача
3. Розробити та використати активні елементи
4. Розробити та використати шаблони не менше 5 шт.
5. Розрахувати швидкодію роботи з інтерфейсом користувача
6. Адаптувати розроблене прикладне програмне забезпечення до різних роздільних здатностей цифрових пристроїв.

Дата видачі « » 2020р. Керівник Шекета В.І.

Завдання отримала Балан А.І.

Міністерство освіти і науки України

Івано-Франківський національний технічний університет нафти та газу

Інститут інформаційних технологій

ЗАТВЕРДЖУЮ

зав. кафедри ІПЗ, проф., д.т.н.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ **В.І. Шекета**

«\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2021р.

ТЕХНІЧНЕ ЗАВДАННЯ

На розробку прикладного програмного забезпечення з моделювання та аналізу програмного забезпечення»

1. Область застосування — проєктування інформаційних систем .
2. Основа розробки — робочий навчальний план дисципліни.
3. Мета та експлуатаційне призначення:
4. мета – отримання практичних навичок проєктування та конфі-

гурування інформаційних систем ;

1. призначення розробки — навчальний курсовий проект із дисци-

пліни «Проєктування інформаційних систем»;

1. Джерела розробки — індивідуальне завдання на курсовий проєкт із дисципліни, технічні рекомендації щодо проектування інформаційних систем та інші технічні матеріали для налаштування окремих компонентів програмної системи.
2. Технічні вимоги

Кінцевий термін виконання курсового проєкту «\_\_\_\_\_\_» 2020 р. Початок розробки «\_\_\_\_\_\_» 2020 р

Порядок контролю та прийняття.

* 1. Виконання етапів технічної та розрахункової документації курсового проєкту, а також моделювання роботи інформаційної системи контролюється викладачем згідно з графіком виконання проєкту;
  2. Прийняття проекту здійснюється комісією, затвердженою зав. кафедри згідно графіку захисту.
  3. Коригування технічного завдання допускається з дозволу керівника проекту.

Розробила студентка групи ІП-20-2 Балан А.І.

ЗМІСТ

[ВСТУП 8](#_Toc74263468)

[1. ОСНОВНА ТЕХНІЧНА ЧАСТИНА 10](#_Toc74263469)

[1.1 Опис сучасного стану динамічних та статичних властивостей предметної області. 10](#_Toc74263470)

[1.2 Опис об’єктів в предметній області згідно прийнятої стилістики. 13](#_Toc74263471)

[1.3 Аналіз сучасних тенденцій до оформлення проекту з проектування інформаційних систем 16](#_Toc74263472)

[1.4 Положення меню 18](#_Toc74263473)

[1.5 Розміщення активних та пасивних елементів на робочому столі інформаційної системи 27](#_Toc74263474)

[1.7 Особливості розробки активних програмних елементів 38](#_Toc74263475)

[1.8 Особливості розробки активних програмних шаблонів 41](#_Toc74263476)

[1.9 Розрахунок якості інтерфейсу користувача 42](#_Toc74263477)

[1.10 Розрахунок швидкості заповнення діалогових форм GOMS 44](#_Toc74263478)

[1.11 Моделювання та тестування роботи інформаційної системи 45](#_Toc74263479)

[ВИСНОВКИ 47](#_Toc74263480)

[ВИКОРИСТАНІ ДЖЕРЕЛА ІНФОРМАЦІЇ 48](#_Toc74263481)

[Додаток А 49](#_Toc74263482)

[Додаток Б 56](#_Toc74263483)

АНОТАЦІЯ

У курсовій роботі вивчено предметну область “Залізничної пасажирської станції” та проаналізовано необхідність побудови автоматизованої системи для успішного документообігу залізничної пасажирської станції.

Метою роботи є дослідження проблем побудови та реалізація компонентів автоматизованої системи: бази даних та інтерфейсу користувача для взаємодії з серверною частиною.

При виконанні роботи було досліджено можливі інструменти та шляхи вирішення даної проблеми.

Результатом роботи є побудовані компоненти системи, опис процесу розробки цих компонентів: реалізована база даних для збереження необхідної інформації, побудовано інтерфейс користувача для керування системою.

Розробка бази даних здійснювалася засобами CodeBlocks.

ВСТУП

С++ є результатом еволюції мови С , яка на початку 70-х років набула широкої популярності. Сьогодні більшість операційних систем написано мовою С++. Порівняно з С, мова С++ набула багато “прикрас”, головною з яких є можливість об’єктно-орієнтованого програмування. Об’єкти за своєю суттю є програмними компонентами, які моделюють предмети реального світу. Модульний, об’єктно-орієнтований підхід до проектування та реалізації програмного забезпечення дозволив полегшити написання складних програмних проектів. Об’єктно-орієнтовані програми є набагато простіші для розуміння,  виправлення та модифікації.

Оскільки сьогодні технології швидко змінюються, оновлюються та застарівають, рівень новизни та складності ПЗ є дуже високим з’являється необхідність в технології проектування програмного забезпечення

Технологія проектування (розроблення) програмного забезпечення (ТППЗ) – це комплекс організаційних заходів, операцій та прийомів, спрямованих на розроблення програмних продуктів високої якості в рамках відведеного бюджету і в термін.

Метою процесу оцінювання ТППЗ є визначення функціональності та якості ТППЗ для наступного вибору. Оцінювання виконується відповідно до конкретних критеріїв, його результати містять як об’єктивні, так і суб’єктивні дані по кожній ТППЗ. Коли аналізуються кінцеві результати оцінювання та до них застосовуються критерії вибору, то може бути рекомендовано придбання технології. Альтернативою може стати відсутність адекватної технології, в такому випадку рекомендується

розробити нову технологію, модифікувати існуючу або відмовитись від впровадження.

Актуальність: створення бази даних передбачає вирішення цілого комплексу проблем, в тому числі розробку програмних, мовних, організаційних і технічних засобів, призначених для централізованого зберігання і колективного використання інформації.

На прикладі опису діяльності інформаційного центру залізничної пасажирської станції та функціонування його бази виконано об’єктний аналіз завдання, розроблено ієрархію класів й алгоритми взаємодії їх об’єктів, складено програму на мові С++, яка реалізовує управління базою даних інформаційного центру залізничної пасажирської станції в консольному інтерфейсі.

Мета: виконати моделювання та аналіз АІС залізничної пасажирської станції та розробити програму для виконання базових запитів для цього підприємства.

Об’єктом дослідження є розробка програмного забезпечення автоматизованої системи залізничної пасажирської станції.

Предмет дослідження: реалізація мовою С++ об’єктно-класової моделі.

Теоретико-методологічною основою розробки є парадигма ООП.

Робота є рукописом об’ємом 70 сторінок машинописного тексту, список використаних джерел на 4 позицій, містить додатки, в тому числі код програми та результати виконання контрольного прикладу.

1. ОСНОВНА ТЕХНІЧНА ЧАСТИНА

1.1 Опис сучасного стану динамічних та статичних властивостей предметної області.

Предметною областю називаються елементи матеріальної системи, інформація про які зберігається і обробляється в ЕІС «Електронній інформаційній системі». Предметна область - сукупність пов'язаних між собою функцій, завдань управління, за допомогою яких досягається виконання поставлених цілей, це частина реального світу, що представляє інтерес для конкретного дослідження.

Об’єкт дослідження - предметна область «Залізнична пасажирська станція». Для аналізу вибрано такі поїзди, яким властиві базові характеристики та атрибути залізничної пасажирської станції . Основним завданням всіх залізничних установ є отримання населенням транспортних послуг. Детальніші характеристики залежать, зокрема, від типу залізничної станції. Розглянемо основні види транспортних засобів, які будуть присутніми в даній курсовій роботі, та їх функції. Отже, основними різновидами, які ми будемо розглядати, таких транспортних засобів є:

* Поїзд(вагони) – транспортний засіб, який вміщає певну кількість пасажирів, для їхнього комфортного пересування
* Локомотив – транспортний засіб, який рухає вагони, переміщує людей від початкової точки до кінцевої;

Також потрібно зауважити, що кожен вагон(їх є 3 типи) вміщує різну кількість людей, і також від кількості людей буде залежати ціна квитка

Працівників залізничної станції можна підрозділити на водіїв рухомого складу, диспетчерів, ремонтників рухомого складу, касирів. Відділи очолюються начальниками, які представляють собою адміністрацію залізничної станції.

Водії локомотивів зобов'язані проходити кожен рік медогляд. Тим, які не пройшли медогляд, необхідно перевестися на іншу роботу. Локомотив повинен своєчасно оглядатися техніками-ремонтниками і при необхідності ремонтуватися. Підготовка до рейсу включає в себе технічну частину (рейсовий техогляд, дрібний ремонт).

Метою будь-якої інформаційної системи є обробка даних про об'єкти реального світу. У звуженому сенсі база даних - це сукупність відомостей про об'єкти залізничної пасажирської станції. Ця система дозволяє виводити дані про пасажирів, та обслуговуючий персонал, інформацію про квитки, рейси, поїзди

У зазначеному випадку головними користувачами наведеної системи буде адміністрація залізничної станції. Пасажири не можуть використовувати дану інформаційну систему. Адміністрація повністю контролює дані цієї інформаційної системи та несе відповідальність за збереження, обробку, передання адміністративної інформації пасажирів, персоналу та станції.

Розроблена інформаційна система призначена для надання та обробки наступних запитів:

Види запитів в інформаційній системі:

1. Отримати перелік і загальне число всіх працівників залізничної станції, начальників відділів, працівників вказаного відділу, за стажем роботи на станції, статевою ознакою, віком, ознакою наявності та кількості дітей, розміру заробітної плати.
2. Отримати перелік і загальне число водіїв локомотивів, які пройшли медогляд або не пройшли медогляд у вказаний рік, за статевою ознакою, віком, розміром заробітної плати.
3. Отримати перелік і загальне число локомотивів, приписаних до залізничної станції, що знаходяться на ній в зазначений час, за часом прибуття на станції, за кількістю скоєних маршрутів, що пройшли плановий техогляд за певний період часу, відправлених у ремонт в означений час, ремонтованих вказане число разів, за кількістю скоєних рейсів до ремонту, за віком локомотива.
4. Отримати перелік і загальне число поїздів на зазначеному маршруті, по тривалості маршруту, за ціною квитка.
5. Отримати перелік і загальне число скасованих рейсів повністю, у зазначеному напрямку, за вказаним маршрутом.
6. Отримати перелік і загальне число затриманих рейсів повністю, за вказаним маршрутом, і кількість зданих квитків за час затримки.
7. Отримати перелік і загальне кількість проданих квитків за вказаний інтервал часу на певні маршрути, по тривалості маршруту, за ціною квитка.

1. Отримати перелік і загальне число пасажирів на зазначеному рейсі, які виїхали в зазначений день, які виїхали за кордон у зазначений день, за ознакою здачі речей в багажне відділення, за статевою ознакою, за віком.
2. Отримати перелік і загальне число невикуплених квитків на зазначеному рейсі, дні.
3. Отримати перелік і загальне число зданих квитків на вказаний рейс, день.

Дана інформаційна система надає доступ адміністрації залізничної станції полегшити процес пошуку потрібної інформації в єдиній базі даних.

1.2 Опис об’єктів в предметній області згідно прийнятої стилістики.

Побудова бази даних, в більшій мірі, ґрунтується на виділенні об'єктів предметної області та структуризації документів, де ці об'єкти, ґрунтуючись на логічному аналізі атрибутів, можна виділити. Зазвичай об'єкти предметної області явно представлені в предметної. Якщо з матеріальними об'єктами все досить очевидно, то документарні об'єкти не завжди можна відразу побачити.

Під документарних об'єктом розуміється елемент, над яким проводяться дії учасниками предметної області, що представляється самим документом або його складовою неподільною частиною, яку можна виділити в самостійний документ.

Однак, виділення документарного об'єкта, представленого у формі документа, містить певну небезпеку - розгляд об'єкта "Документ". При такому розгляді інформаційна система стає реалізацією системи документообігу, а база даних виконується у вигляді сховища документів, що для вирішення задач облікового характеру, що не передбачає реалізації системи документообігу, буде невірним. Щоб уникнути такої небезпеки, необхідно документарний об'єкт розглядати не у вигляді об'єкта "Документ", а представляти його функціональним об'єктом предметної області.

В цьому проекті ключовими (тобто відомості про яких становлять основу діяльності в предметній області) об’єктами є:

* Працівники;
* Пасажири;
* Поїзди;
* Локомотиви;
* Квитки;
* Рейси.

У деяких випадках об'єкти функції є ідентичними, що володіють загальним змістом і суттю, що дозволяє об'єднати ці об'єкти в єдиний об'єкт з виділенням категорій. Тобто в такому разі необхідно виконати операцію категоризації об'єктів предметної області для відповідної функції.

Під категоризацією об'єктів розуміється процес узагальнення сукупності об'єктів предметної області, що володіють єдиним сенсом і загальним набором атрибутів, з виділенням розглянутих об'єктів в якості об'єктів-категорій, які подаються на більш низьких рівнях деталізації предметної області.

Категоризація об'єктів предметної області є одним з варіантів їх класифікації, яка при аналізі предметної області дозволяє істотно знизити кількість розглянутих об'єктів і спростити саму предметну область для реалізації її структур даних.

Класифікація об'єктів, яка групує об'єкти предметної області, визначається на підставі декількох принципів, що дозволяють проводити операції угруповання над об'єктами предметної області:

• в один клас включаються об'єкти предметної області з загальними характеристиками і властивостями;

• в один клас об'єктів входить не менше двох об'єктів;

• об'єкти предметної області можуть мати ієрархічну залежність.

Ключові об’єкти предметної області «Залізнична пасажирська станція»:

* Об’єкт «Працівники»:

Атрибути, які відповідають об’єкту: прізвище, посада, досвід, стать, вік, ознака наявності дітей (та їх кількості ), розміру заробітної плати, ознака проходження медогляду.

* Об’єкт «Пасажири»:

Атрибути, які відповідають об’єкту: прізвище, тип вагону, в якому буде їхати пасажир, ознака від’їзду в зазначений день, ознака від’їзду за кордон в зазначений день, ознакою здачі речей в багажне відділення, стать, вік .

* Об’єкт «Поїзди»:

Атрибути, які відповідають об’єкту: назва, маршрут, тривалість, ціна квитка.

* Об’єкт «Локомотиви»:

Атрибути, які відповідають об’єкту: назва, ознака знаходження локомотиву на станції, час прибуття, кількість скоєних маршрутів, ознака проходження планового техогляду, ознака відправлених у ремонт в означений час, кількість ремонтів, кількість скоєних маршрутів до ремонту, вік.

* Об’єкт «Квитки»:

Атрибути, які відповідають об’єкту: назва, тип квитка(проданий, повернутий, невикуплений), дата відправлення, тривалість, тип поїзда.

* Об’єкт «Рейси»:

Атрибути, які відповідають об’єкту: назва, тип рейсу ( затриманий, скасований), тип поїзда, вік.

1.3 Аналіз сучасних тенденцій до оформлення проекту з проектування інформаційних систем

Інформаційна система, як система управління, тісно пов’язується, як системами збереження та видачі інформації, так і з іншої - з системами, що забезпечують обмін інформацією в процесі управління. Вона охоплює сукупність засобів та методів, що дозволяють користувачу збирати, зберігати, передавати і обробляти відібрану інформацію.

Інформаційна система (ІС) — це взаємопов'язана сукупність пристроїв, методів і персоналу для обробки інформації.

Інформаційні системи існують з моменту появи суспільства, оскільки на кожній стадії його розвитку існує потреба в управлінні. Місією інформаційної системи є виробництво потрібної для організації інформації, потрібної для ефективного управління всіма її ресурсами, створення інформаційного та технічного середовища для управління її діяльністю.

Автоматизована інформаційна система – це взаємозв’язана сукупність даних, обладнання, програмних засобів, персоналу, стандартних процедур, які призначені для збору, обробки, розподілу, зберігання, представлення інформації у відповідності з вимогами, які випливають з цілей організації. Сьогодні, у вік інформації, практично кожна інформаційна система використовує комп’ютерні технології, і тому надалі під інформаційними системами надалі будемо підрозумівати саме автоматизовані . Саме ці системи дозволяють швидко і надійно обробляти запити і інформацію, тим самим полегшуючи життя людині.

Інформаційні системи включають в себе: технічні засоби обробки даних, програмне забезпечення і відповідний персонал. Чотири складові частини

* утворюють внутрішню інформаційну основу:
* засоби фіксації і збору інформації;
* засоби передачі відповідних даних та повідомлень;
* засоби збереження інформації;
* засоби аналізу, обробки і представлення інформації.

Аналіз сучасного стану ринку ІС показує стійку тенденцію зростання попиту на інформаційні системи організаційного управління.

1.4 Положення меню

Меню - елемент інтерфейсу користувача, що дозволяє вибрати одну з декількох перерахованих опцій програми, які, в свою чергу, теж мають перераховані опції. Усе залежить від коду програми. У сучасних операційних системах найважливішим елементом графічного інтерфейсу користувача є меню .

Пункти меню можуть вибиратися користувачем будь-яким з пристроїв введення.

Розрізняють такі типи меню:

1. за виконанням:

* текстове;
* графічне.

1. за функціями:

* головне меню програми;
* спливаюче меню;
* контекстне меню;
* системне меню.

У даній курсовій роботів створено головне меню, яке містить 11 функцій. Кожна з цих функцій містить спливаюче меню, з функціями посортованих по певній запиту. При виборі одного з пунктів спливаючого вікна, користувач зможе одержати результат пошуку.

Меню створеної консольної програми складається з 11 пунктів та має такий вигляд:

void MainMenu () {

"Menu "

"(0) Exit the program "

"(1) Get a list and the total number of employees of the railway station"

"(2) Get a list and total number of locomotive drivers "

"(3) Get a list and total number of locomotives "

"(4) Get the list and total number of trains "

"(5) Get the list and the total number of canceled flights in full "

"(6) Get the list and total number of detained flights in full "

"(7) Get a list and total number of tickets sold "

"(8) Get a list and the total number of passengers "

"(9) Get a list and the total number of unredeemed tickets "

"(10) Get a list and the total number of returned tickets "

"Your choice "

Ці пункти охоплюють велику кількість інформації, тому потрібно описати кожен з них.

Пункт 0: «Exit the program»

Це один з головних пунктів меню, за допомогою якого користувач може закінчити роботу програми. Для того щоб закінчити виконання програми потрібно натиснути “0” і натиснути будь-яку клавішу для виходу.

Пункт 1: «Get a list and the total number of employees of the railway station»

При ввід на екран символу “1” консольное меню очищується і з’являється спливаюче меню, яке подано нижче.

"Menu "

"Get a list and the total number of employees of the railway station \n"

"(0) Exit the program "

"(1) all employees "

"(2) department heads"

"(3) employees of the specified department "

"(4) by length of service at the station "

"(5) gender "

"(6) age "

"(7) a sign of the presence and number of children "

"(8) the size of the salary "

"Your choice "

Дане меню дозволяє отримати перелік і загальне число всіх працівників залізничної станції (символ “1”), начальників відділів(символ “2”), працівників вказаного відділу(символ “3”), за стажем роботи на станції(символ “4”), статевою ознакою(символ “5”), віком(символ “6”), ознакою наявності та кількості дітей(символ “7”), розміру заробітної плати(символ “8”). Вийти з програми можна натиснувши символ “0”).

Після вибору певного запиту спливаючого меню, користувачу буде відображено перелік пунктів, з яких можна вибрати необхідний для нього. Після отримання програмою введеного значення викликається необхідна функція виводу і користувач отримає результат.

Пункт 2: «Get a list and total number of locomotive drivers»

При ввід на екран символу “2” консольное меню очищується і з’являється спливаюче меню, яке подано нижче.

"Menu \n"

"Get a list and the total number of locomotive drivers "

"(0) Exit the program "

"(1) for medical examination "

"(2) by gender "

"(3) age "

"(4) the amount of salary "

"Your choice "

Дане меню дозволяє отримати перелік і загальне число водіїв локомотивів, які пройшли медогляд або не пройшли медогляд у вказаний рік (символ “1”),за статевою ознакою (символ “2”), віком (символ “3”), розміром заробітної плати (символ “4”). Вийти з програми можна натиснувши символ “0”).

Після вибору певного запиту спливаючого меню, користувачу буде відображено перелік пунктів, з яких можна вибрати необхідний для нього. Після отримання програмою введеного значення викликається необхідна функція виводу і користувач отримає результат. Кожен запит відрізняється своїм описом, це дозволило розділити і помістити його в декілька пунктів.

Пункт 3: «Get a list and total number of locomotives»

При ввід на екран символу “3” консольное меню очищується і з’являється спливаюче меню, яке подано нижче.

"Menu "

"Get a list and total number of locomotives "

"(0) Exit the program "

"(1) assigned to the railway station "

"(2) at the time of arrival at the station "

"(3) by number of routes completed "

"(4) that have passed the scheduled inspection "

"(5) sent for repair "

"(6) repaired "

"(7) by number of routes completed before repair "

"(8) by age "

"Your choice "

Отримати перелік і загальне число водіїв локомотивів, які пройшли медогляд або не пройшли медогляд у вказаний рік (символ “1”), за статевою ознакою (символ “2”), віком (символ “3”), розміром заробітної плати (символ “4”). Вийти з програми можна натиснувши символ “0”).

Після вибору певного запиту спливаючого меню, користувачу буде відображено перелік пунктів, з яких можна вибрати необхідний для нього. Кожен запит, має унікальні пункти, характерні тільки для нього. Після отримання програмою введеного значення викликається необхідна функція виводу і користувач отримає результат. Кожен запит відрізняється своїм описом, це дозволило розділити і помістити його в декілька пунктів.

Пункт 4: «Get the list and total number of trains»

При ввід на екран символу “4” консольное меню очищується і з’являється спливаюче меню, яке подано нижче.

"Menu "

"Get list and total number of trains "

"(0) Exit the program "

"(1) on the specified route "

"(2) by route duration "

"(3) at the ticket price "

"Your choice ";

Отримати перелік і загальне число поїздів на зазначеному маршруті (символ “1”), по тривалості маршруту (символ “2”), за ціною квитка (символ “3”). Вийти з програми можна натиснувши символ “0”).

При вводі потрібного тексту користувач має звертати увагу на правильність їхнього запиту, оскільки від цього буде залежати результат роботи.

Пункт 5: «Get the list and total number of canceled flights in full»

При ввід на екран символу “5” консольное меню очищується і з’являється спливаюче меню, яке подано нижче.

"Menu "

"Get the list and the total number of canceled flights in full "

"(0) Exit the program "

"(1) on the specified route "

"Your choice \n";

Отримати перелік і загальне число скасованих рейсів повністю, за вказаним маршрутом (символ “1”). Вийти з програми можна натиснувши символ “0”).

В даному випадку наявний тільки 1 запит, який користувач може ввести самостійно. Користувачу буде представлений перелік маршрутів, один з яких потрібно обрати. Користувач правильно вводить вибраний ним маршрут і отримує правильний результат.

Пункт 6: «Get the list and total number of detained flights in full»

При ввід на екран символу “6” консольное меню очищується і з’являється спливаюче меню, яке подано нижче.

"Menu "

"Get the list and total number of detained flights in full "

"(0) Exit the program "

"(1) on the specified route "

"(2) the number of tickets handed over during the detain "

"Your choice "

Отримати перелік і загальне число затриманих рейсів повністю, за вказаним маршрутом (символ “1”), і кількість зданих квитків за час затримки (символ “2”). Вийти з програми можна натиснувши символ “0”).

В даному випадку подано тільки 2 запит, які користувач може ввести самостійно. В 1 запиті користувачу буде представлений перелік маршрутів, один з яких потрібно обрати. В 2 запиті користувачу буде представлений перелік кількості зданих квитків за час затримки, один з яких потрібно обрати. Користувач правильно вводить вибрані ним дані і отримує правильний результат.

Пункт 7: «Get a list and total number of tickets sold»

При ввід на екран символу “7” консольное меню очищується і з’являється спливаюче меню, яке подано нижче.

"Menu "

"Get a list and total number of tickets sold "

"(0) Exit the program "

"(1) on the specified route "

"(2) by route duration"

"(3) at the ticket price "

"Your choice "

Отримати перелік і загальне кількість проданих квитків за вказаний інтервал часу на певні маршрути (символ “1”), по тривалості маршруту (символ “2”), за ціною квитка. (символ “3”), Вийти з програми можна натиснувши символ “0”).

В даному випадку подано тільки 3 запит, які користувач може ввести самостійно. В 1 запиті користувачу буде представлений перелік маршрутів, один з яких потрібно обрати. В 2 запиті користувачу буде представлений перелік годин, скільки тривають маршрути, один з яких потрібно обрати. . В 3 запиті користувачу необхідно буде вибрати одну з запропонованих цін квитка. Користувач правильно вводить вибрані ним дані і отримує правильний результат.

Пункт 8: «Get a list and the total number of passengers»

При ввід на екран символу “8” консольное меню очищується і з’являється спливаюче меню, яке подано нижче.

"Menu "

"Get list and total number of passengers "

"(0) Exit the program "

"(1) who left on the specified route "

"(2) who left on the specified day "

"(3) who went abroad on a given day "

"(4) on the basis of putting things in the luggage compartment "

"(5) by gender "

"(6) by age"

"Your choice "

Отримати перелік і загальне число пасажирів на зазначеному рейсі (символ “1”), які виїхали в зазначений день (символ “2”), які виїхали за кордон у зазначений день (символ “3”), за ознакою здачі речей в багажне відділення (символ “4”), за статевою ознакою (символ “5”), за віком (символ “6”). Вийти з програми можна натиснувши символ “0”).

Даний запит складається з 6 пунктів, кожен з яких характеризується іншими даними і результатами. Користувач обирає будь-який з цих запитів, отримує перелік з якого можна обрати, правильно вводить дані і одержує результат.

Пункт 9: «Get a list and the total number of unredeemed tickets»

При ввід на екран символу “9” консольное меню очищується і з’являється спливаюче меню, яке подано нижче.

"Menu "

"Get list and total number of unredeemed tickets"

"(0) Exit the program "

"(1) on the specified flight "

"(2) on the specified day "

"Your choice "

У даному запиті наявні тільки 2 пункти, які користувач може ввести самостійно. В 1 запиті користувачу буде представлений перелік маршрутів, один з яких потрібно обрати. В 2 запиті користувачу буде представлений перелік дат, одну з яких потрібно обрати. При коректному введені користувач отримує правильний результат.

Пункт 10: «Get a list and the total number of returned tickets»

При ввід на екран символу “10” консольное меню очищується і з’являється спливаюче меню, яке подано нижче.

"Menu "

"Get a list and total number of returned tickets "

"(0) Exit the program "

"(1) on the specified flight "

"(2) on the specified day "

"Your choice "

Даний запит містить 2 пункти, які користувач може ввести самостійно. В 1 запиті користувачу буде представлений перелік маршрутів, один з яких потрібно обрати. В 2 запиті користувачу буде представлений перелік дат, одну з яких потрібно обрати. Користувач правильно вводить вибрані дані і отримує результат.

1.5 Розміщення активних та пасивних елементів на робочому столі інформаційної системи

Інтерфейс користувача – це сукупність засобів, за допомогою яких користувач спілкується з різними пристроями (з комп’ютером або побутовою технікою) або іншим складним інструментарієм (системою). Інтерфейс користувача - це такий різновид інтерфейсів, в якому з одного боку - людина, з іншого - машина (пристрій, програмне забезпечення).

За визначенням Національного банку стандартизованих науково-технічних термінів, інтерфейс користувача - це комплекс апаратних і програмних засобів, що забезпечує взаємодію користувача з комп'ютером. Інтерфейс користувача комп’ютерного додатку включає:

1) засоби відображення інформації, відображувану інформацію, формати і коди;

2) командні режими, мову «користувач–інтерфейс»;

3) пристрої та технології введення-виведення;

4) діалоги, взаємодію та транзакції між користувачем та комп’ютером, зворотній зв’язок з користувачем;

5) підтримку прийняття рішень в конкретній предметній області;

6) порядок використання програми і документації на неї.

Інтерфейс користувача часто розуміють лише як зовнішній вигляд програми. Однак насправді користувач сприймає через нього всю програму в цілому, тобто таке розуміння є надто вузьким.

В даній цій курсовій роботі увага спрямована на структуроване, зрозуміле меню, для того, щоб користувач зміг одразу зорієнтуватися в програмі і почав працювати з нею. Меню представлене блоками, щоб зменшити кількість неструктурованих запитів в меню. Для цього ми помістили всі запити, що відносяться до одного об’єкта в один спільний запит (блок).

1.6 Прикладне програмування інформаційної системи

Для правильного функціонування будь-якої програми необхідно прописати ряд інструкцій, для того, щоб користувач, вперше побачивши програму, зміг швидко зорієнтуватися в ній. Отже, дана інструкція подана нижче.

Інструкція використання даної інформаційної системи:

1. Користувачу потрібно відкрити компілятор CodeBlocks для запуску проєкту. Запускаємо програму, яка прописана в компіляторі, оскільки це є один з найзручніших способів запуску програм.
2. Після запуску користувач побачить текст “Для продовження натисніть будь-яку клавішу…”. Користувач, натиснувши будь-яку кнопку, побачить головне меню, яке складається з 10 пунктів. Користувач ознайомлюється з кожним із запитів і обирає той, що задоволяняє його пошук. Кожен пукт має свій номер поданий в дужках ( наприклад (1)). Користувач набирає на клавіатурі необхідний номер і натискає кнопку Enter . Всі інші запити теж можна відкрити за допомогою певної цифри, яка відповідає поданим запитам.
3. Після обрання 1 пункту, система виводить спливаюче меню, яке містить 8 нових запитів.
4. Враховуючи пункти, які ми обрали:

* Пункт 1:

Отримати перелік і загальне число працівників залізничної станції. Обравши цей пункт, ми отримаємо 8 запитів від системи:

* Запит 1:

Обравши цей запит, система виводить на екран всіх співробітників залізничної станції. Також, система виводить загальну кількість працівників.

* Запит 2:

Обравши цей запит, система виводить на екран список, яких відділів є начальники на даній залізничній станції . Користувач обирає один з поданих варіантів, вводить цю посаду і система виводить всіх начальників вказаного відділу. Також, система виводить загальну кількість працівників.

* Запит 3:

Обравши цей запит, система виводить на екран список, які є відділи на даній залізничній станції. Користувач обирає один з поданих варіантів, вводить цю посаду і система виводить на екран всіх співробітників залізничної станції вказаного відділу. Також, система виводить загальну кількість працівників.

* Запит 4:

Обравши цей запит, система виводить на екран список стажу працівників. Користувач обирає один з поданих варіантів, вводить цей стаж і система виводить всіх працівників, які мають вказаний стаж. Також, система виводить загальну кількість працівників.

* Запит 5:

Обравши цей запит, система виводить на екран список статі працівників. Користувач обирає один з поданих варіантів, вводить цю стать і система виводить всіх працівників даної статі. Також, система виводить загальну кількість працівників.

* Запит 6:

Обравши цей запит, система виводить на екран список віку працівників. Користувач обирає один з поданих варіантів, вводить цей вік і система виводить всіх працівників, які мають вказаний вік. Також, система виводить загальну кількість працівників.

* Запит 7:

Обравши цей запит, система виводить на екран список ознак, за якими працівники можуть мати а можуть і не мати дітей,. Користувач обирає один з поданих варіантів, вводить цю ознаку і система виводить всіх працівників, хто має дітей і їхню кількість, або навпаки. Також, система виводить загальну кількість працівників.

* Запит 8:

Обравши цей запит, система виводить на екран список зарплат працівників. Користувач обирає один з поданих варіантів, вводить цю суму і система виводить всіх працівників, які мають вказану зарплату. Також, система виводить загальну кількість працівників.

* Пункт 2:

Отримати перелік і загальне число водіїв локомотивів. Обравши цей пункт, ми отримаємо 4 запити від системи:

* Запит 1:

Обравши цей запит, система виводить на екран список ознак, за якими водіїв можуть пройти медогляд або не пройти його. Користувач обирає один з поданих варіантів, вводить цю ознаку і система виводить всіх водіїв, хто пройшов медогляд, або навпаки. Також, система виводить загальну кількість водіїв.

* Запит 2:

Обравши цей запит, система виводить на екран список статі водіїв. Користувач обирає один з поданих варіантів, вводить цю стать і система виводить всіх працівників даної статі. Також, система виводить загальну кількість водіїв.

* Запит 3:

Обравши цей запит, система виводить на екран список віку водіїв. Користувач обирає один з поданих варіантів, вводить цей вік і система виводить всіх працівників, які мають вказаний вік. Також, система виводить загальну кількість водіїв.

* Запит 4:

Обравши цей запит, система виводить на екран список зарплат водіїв. Користувач обирає один з поданих варіантів, вводить цю суму і система виводить всіх працівників, які мають вказану зарплату. Також, система виводить загальну кількість водіїв.

* Пункт 3:

Отримати перелік і загальне число локомотивів. Обравши цей пункт, ми отримаємо 8 запитів від системи:

* Запит 1:

Обравши цей запит, система виводить на екран список ознак локомотивів, що знаходяться на станції або ні. Користувач обирає один з поданих варіантів, вводить цю ознаку і система виводить всі локомотиви, що знаходяться на станції, або навпаки. Також, система виводить загальну кількість локомотивів.

* Запит 2:

Обравши цей запит, система виводить на екран список часу прибуття на станції. Користувач обирає один з поданих варіантів, вводить це значення і система виводить всі локомотиви, прибувають на станцію в зазначений час. Також, система виводить загальну кількість локомотивів.

* Запит 3:

Обравши цей запит, система виводить на екран список кількості скоєних рейсів локомотивом. Користувач обирає один з поданих варіантів, вводить це значення і система виводить всі локомотиви, ,за кількістю скоєних рейсів. Також, система виводить загальну кількість локомотивів.

* Запит 4:

Обравши цей запит, система виводить на екран список ознак локомотивів, що пройшли техогляд або ні. Користувач обирає один з поданих варіантів, вводить цю ознаку і система виводить всі локомотиви, що пройшли техогляд, або навпаки. Також, система виводить загальну кількість локомотивів.

* Запит 5:

Обравши цей запит, система виводить на екран список ознак локомотивів, відправлених у ремонт або ні. Користувач обирає один з поданих варіантів, вводить цю ознаку і система виводить всі локомотиви, відправлених у ремонт, або навпаки. Також, система виводить загальну кількість локомотивів.

* Запит 6:

Обравши цей запит, система виводить на екран список ознак локомотивів, що були ремонтовані або ні. Користувач обирає один з поданих варіантів, вводить цю ознаку і система виводить всі локомотиви, що були ремонтовані, або навпаки. Також, система виводить загальну кількість локомотивів.

* Запит 7:

Обравши цей запит, система виводить на екран список кількості скоєних рейсів локомотивом до ремонту. Користувач обирає один з поданих варіантів, вводить це значення і система виводить всі локомотиви, за кількістю скоєних рейсів до ремонту. Також, система виводить загальну кількість локомотивів.

* Запит 8:

Обравши цей запит, система виводить на екран список віку локомотивів . Користувач обирає один з поданих варіантів, вводить це значення і система виводить всі локомотиви певного віку. Також, система виводить загальну кількість локомотивів.

* Пункт 4:

Отримати перелік і загальне число поїздів. Обравши цей пункт, ми отримаємо 3 запити від системи:

* Запит 1

Обравши цей запит, система виводить на екран список маршрутів поїздів. Користувач обирає один з поданих варіантів, вводить цей рейс і система виводить всі поїзди на даний маршрут. Також, система виводить загальну кількість поїздів.

* Запит 2

Обравши цей запит, система виводить на екран список тривалості маршруту. Користувач обирає один з поданих варіантів, вводить цю кількість і система виводить поїзди на даний маршрут. Також, система виводить загальну кількість поїздів.

* Запит 3

Обравши цей запит, система виводить на екран список вартості квитків. Користувач обирає один з поданих варіантів, вводить цю кількість і система виводить всі поїзди за кількістю квитків. Також, система виводить загальну кількість поїздів.

* Пункт 5:

Отримати перелік і загальне число скасованих рейсів повністю. Обравши цей пункт, ми отримаємо 1 запити від системи:

* Запит 1

Обравши цей запит, система виводить на екран список маршрутів поїздів. Користувач обирає один з поданих варіантів, вводить цей рейс і система виводить всі скасованих рейси на даному маршруті. Також, система виводить загальну кількість скасованих рейсів повністю.

* Пункт 6:

Отримати перелік і загальне число затриманих рейсів повністю. Обравши цей пункт, ми отримаємо 2 запити від системи:

* Запит 1

Обравши цей запит, система виводить на екран список маршрутів поїздів. Користувач обирає один з поданих варіантів, вводить цей рейс і система виводить всі затриманих рейси на даному маршруті. Також, система виводить загальну кількість затриманих рейсів повністю.

* Запит 2

Обравши цей запит, система виводить на екран список кількістю зданих квитків за час затримки. Користувач обирає один з поданих варіантів, вводить цю кількість і система виводить всі затриманих рейси за кількістю зданих квитків. Також, система виводить загальну кількість затриманих рейсів повністю.

* Пункт 7:

Отримати перелік і загальне число проданих квитків. Обравши цей пункт, ми отримаємо 3 запити від системи:

* Запит 1

Обравши цей запит, система виводить на екран список маршрутів поїздів. Користувач обирає один з поданих варіантів, вводить цей рейс і система виводить всі продані квитки на даний маршрут. Також, система виводить загальну кількість проданих квитків.

* Запит 2

Обравши цей запит, система виводить на екран список тривалості маршруту. Користувач обирає один з поданих варіантів, вводить цю кількість і система виводить всі продані квитки на даний маршрут. Також, система виводить загальну кількість проданих квитків.

* Запит 3

Обравши цей запит, система виводить на екран список вартості проданих квитків. Користувач обирає один з поданих варіантів, вводить цю кількість і система виводить всі продані квитки за вказаною вартістю. Також, система виводить загальну кількість проданих квитків.

* Пункт 8:

Отримати перелік і загальне число пасажирів. Обравши цей пункт, ми отримаємо 6 запитів від системи:

* Запит 1:

Обравши цей запит, система виводить на екран список маршрутів поїздів. Користувач обирає один з поданих варіантів, вводить цей рейс і система виводить всіх пасажирів на зазначеному рейсі. Також, система виводить загальну кількість пасажирів.

* Запит 2:

Обравши цей запит, система виводить на екран список ознак пасажирів, які виїхали в зазаначений день або ні. Користувач обирає один з поданих варіантів, вводить цю ознаку і система виводить всіх пасажирів, які виїхали в зазаначений день, або навпаки. Також, система виводить загальну кількість пасажирів.

* Запит 3:

Обравши цей запит, система виводить на екран список ознак пасажирів, які виїхали в зазаначений день за кордон або ні. Користувач обирає один з поданих варіантів, вводить цю ознаку і система виводить всіх пасажирів, які виїхали в зазаначений день за кордон, або навпаки. Також, система виводить загальну кількість пасажирів.

* Запит 4:

Обравши цей запит, система виводить на екран список ознак, за який пасажир, може здати речі в багажне відділення або ні. Користувач обирає один з поданих варіантів, вводить цю ознаку і система виводить всіх пасажирів, хто здав речі в багажне відділення, або навпаки. Також, система виводить загальну кількість пасажирів.

* Запит 5:

Обравши цей запит, система виводить на екран список статі пасажирів. Користувач обирає один з поданих варіантів, вводить цю стать і система виводить всіх пасажирів даної статі. Також, система виводить загальну кількість пасажирів.

* Запит 6:

Обравши цей запит, система виводить на екран список віку пасажирів. Користувач обирає один з поданих варіантів, вводить цей вік і система виводить всіх пасажирів, які мають вказаний вік. Також, система виводить загальну кількість пасажирів.

* Пункти 9:

Отримати перелік і загальне число невикуплених квитків. Обравши цей пункт, ми отримаємо 2 запити від системи:

* Запит 1:

Обравши цей запит, система виводить на екран список рейсів поїздів. Користувач обирає один з поданих варіантів, вводить цей рейс і система виводить всі поїзди на даному рейсі. Також, система виводить загальну кількість невикуплених квитків.

* Запит 2:

Обравши цей запит, система виводить на екран список дат поїздів. Користувач обирає один з поданих варіантів, вводить цю дату і система виводить всі поїзди в цей період . Також, система виводить загальну кількість невикуплених квитків.

* Пункт 10:

Отримати перелік і загальне число зданих квитків. Обравши цей пункт, ми отримаємо 2 запити від системи:

* Запит 1:

Обравши цей запит, система виводить на екран список рейсів поїздів. Користувач обирає один з поданих варіантів, вводить цей рейс і система виводить всі поїзди на даному рейсі. Також, система виводить загальну зданих квитків.

* Запит 2:

Обравши цей запит, система виводить на екран список дат поїздів. Користувач обирає один з поданих варіантів, вводить цю дату і система виводить всі поїзди в цей період . Також, система виводить загальну зданих квитків.

1.7 Особливості розробки активних програмних елементів

В основі об'єктно-орієнтованої мови програмування лежать два основних поняття: клас та об'єкт. Об'єкт – це базове поняття в ООП, це конкретна реалізація, екземпляр класу. Об'єкт складається з трьох частин: стан (змінні стану), методи (операції), ім'я об'єкта. Клас - це група даних і методів або функцій для роботи з цими даними, це шаблон. Об'єкти однаковими наборами змінних стану і методів, утворюють клас. Якщо об'єкти мають реалізацію з конкретного світу, то класи є абстракціями. Трохи більш складні об'єкти можуть взагалі не містити даних, а представляти процес і містити тільки функції, які реалізують цей процес. Для формування реального об'єкта необхідно мати шаблон, по прикладу якого і будується даний об'єкт.

Поняттю "об'єкт" зіставляють ряд доповнюючих один одного визначень:

* це сукупність змінних стану і пов'язаних з ними методів (операцій). Ці методи визначають як об'єкт взаємодіє з оточуючим його світом;
* це конкретна реалізація, екземпляр класу. У програмуванні відносини об'єкта і класу можна порівняти з описом змінної, де сама змінна (об'єкт) є екземпляром якого-небудь типу даних (класу);
* це відчутна реальність, що характеризується чітко визначеною поведінкою;
* особливий упізнаваний предмет, блок або сутність (реальний чи абстрактний), що має важливе функціональне призначення в даній предметній області.

Є й інша класифікація методів об'єкта, коли виділяють функції управління, реалізації, доступу та допоміжні функції. Індивідуальність об'єкта - це властивість об'єкта, що відрізняє цей об'єкт від усіх інших об'єктів. Об'єкти можуть перебувати в певних відносинах один до одного, відносини можуть бути ієрархічними. Основні ієрархічні відносини називають відносини використання та включення.

Основні властивості об'єктів:

Зазвичай вважається, що без інкапсуляції неможливо уявити собі об’єктно-орієнтоване програмування. Інкапсуляція це здатність об'єктів скривати деякі способи обробки даних (методи) та самі дані від навколишнього цифрового середовища. Історія розвитку методологій програмування спонукувана боротьбою зі складністю розробки програмного забезпечення. Складність великих програмних систем, у створенні яких бере участь відразу велика кількість розробників, зменшується, якщо на верхньому рівні не видно деталей реалізації нижніх рівнів. Власне, процедурний підхід був першим кроком на цьому шляху. Під інкапсуляцією (слово "обволікання") розуміється приховування інформації про внутрішній устрій об'єкта, при якому робота з об'єктом може вестися тільки через його загальнодоступний (public) інтерфейс. Або це механізм програмування, об'єднуючий разом код і дані, якими він маніпулює, виключаючи як втручання ззовні, так і неправильне використання даних. В об'єктно-орієнтованій мові програмування дані і код можуть бути об'єднані в абсолютно автономний чорний ящик. Усередині такого ящика знаходяться всі необхідні дані і код. Коли код і дані зв'язуються разом подібним чином, створюється об'єкт. Іншими словами, об'єкт - це елемент, що підтримує інкапсуляцію.

Поліморфізм - також одна з трьох основних парадигм ООП. Якщо говорити коротко, поліморфізм - це здатність об'єкта використовувати методи похідного класу, який не існує на момент створення базового. Слово «поліморфізм» можна перекласти як «багато форм». Тобто це можливість використання одного і того ж імені операції або методу до об'єктів різних класів, при цьому дії, що здійснюються з об'єктами, можуть істотно різнитися. Тому можна сказати, що в одного і того ж слова багато форм.

Остання парадигма спадкування (inheritance) - стосується здатності мови дозволяти будувати нові визначення класів на основі визначень існуючих класів. По суті, спадкування дозволяє розширювати поведінку базового класу, наслідуючи основну функціональність у похідному підкласі. Спадкування являє собою процес, в ході якого один об'єкт набуває властивостей іншого об'єкта. Це дуже важливий процес, оскільки він забезпечує принцип ієрархічної класифікації. Якщо вдуматися, то велика частина знань піддається систематизації завдяки ієрархічної класифікації по низхідній.

Об’єкти класів є активними програмними елементами даної роботи. Саме тому в ній наявні класи, які є спадкоємцями інших класів. Всі змінні даних класів є інкапсульовані для того, щоб не траплялася помилка у виконанні роботи. Кожен клас міститиме особливі для себе значення

1.8 Особливості розробки активних програмних шаблонів

У процесі розв’язання багатьох задач необхідно використовувати функції, в яких алгоритм обчислення однаковий, а типи даних відрізняються. Прикладом є задачі пошуку і сортування. Особливістю програмування таких задач мовою С++ є використання шаблонів функцій.

Шаблони функцій — потужний засіб параметризації. За до­помогою шаблона функції можна визначити алгоритм, який буде застосовуватися до даних різних типів, а конкретний тип даних передається функції у вигляді параметра на етапі компіляції.

Шаблон функції — це деяка узагальнена функція (родова функція) для сімейства функцій, призначених для розв’язання даної задачі. Визначається така шаблонна функція у заголовочному файлі і має такий вигляд.

Шаблони проєктування програмного забезпечення— ефектні способи вирішення задач проєктування програмного забезпечення. Шаблон не є закінченим зразком, який можна безпосередньо транслювати в програмний код. Об'єктно-орієнтований шаблон найчастіше є зразком вирішення проблеми і відображає відношення між класами та об'єктами, без вказівки на те, як буде зрештою реалізоване це відношення.

Під час виконання даної роботи я використовувала активні програмні шаблони, такі як табуляцію, для раціональнішого і зрозумілішого виставлення об’єктів в рядок.

Отже, для того щоб покращитий роботу системи не потрібно писати безліч функцій, потрібно використовувати потрібні в той момент активні програмні шаблони.

1.9 Розрахунок якості інтерфейсу користувача

Кількісні методи допомагають звести спірні питання в оцінці якості інтерфейсу до простим розрахункам.

Одним з кращих підходів до кількісного аналізу моделей інтерфейсів є класична модель GOMS (the model of goals, objects, methods and selection rules) - модель цілей, об'єктів, методів і вибір правил.

Ця модель заснована на оцінці швидкості друку.

Час, необхідний для виконання якогось завдання системою користувач-комп'ютер, є сумою всіх тимчасових інтервалів, які потрібні були системи на виконання елементарних жестів, що складають дану задачу.

Лабораторним шляхом встановлено стандартні середні інтервали для деяких жестів, виконуваних різними користувачами:

К = 0,2 с - натискання клавіші;

Р = 1,1 с - вказівка ​​(на якусь позицію на екрані монітора);

Н = 0,4 с - переміщення (руки з клавіатури на ГУВ або з ГУВ на клавіатуру;

М = 1,35 с - ментальна;

R - відповідь (час очікування відповіді комп'ютера).

Тепер складемо оператори і обчислимо суму часів:

Н + М + Р + К + Н + М + К + К + К + К + М + К = 0,4 +1,35 +1,1 +0,2 +0,4 +1,35 + (4х0 , 2) +1,35 +0,2 = 7,15 с.

Показники зручності використання інтерфейсу

* Вивчаймість
* Швидкість роботи
* Стійкість
* Відновлюваність
* Адаптується

Для комфортного користування програмою користувачем, я вирішила не змінювати колір меню, оскільки чорний фон і білі літери є оптимальний варіант для сприйняття інтерфейсу.

Також за допомогою символів «|» та «\_» створила рамки, які дозволяють легше сприйняти написаний текст.

Головна мова програми - англійська, оскільки це одна з найпопулярніших мов у світ. Користувач, який не володіє іншою мовою крім англійської з легкістю зможе зрозуміти програми. Саме тому, я вибрала її для даної програми, оскільки ця мова є універсальною для всіх.

Структурований вигляд меню відіграє важливу роль в програмі, оскільки користувач, не розуміючи інтерфейс, не зможе знайти потрібну інформацію, або надіслати певний запит, Для вдалого розприділення запитів в меню я використовувала функцію “\n”, яка переміщує курсор на рядок нижче.

1.10 Розрахунок швидкості заповнення діалогових форм GOMS

Одним з кращих підходів до кількісного аналізу моделей інтерфейсів є класична модель GOMS - «правила для цілей, об'єктів, методів і виділення» (the model of goals, objects, methods and selection rules).

Один найпростіший, але досить цінний аспект методу GOMS - модель, заснована на оцінці швидкості друку. Розробники, які знайомі з методом GOMS, рідко проводять детальний і формальний аналіз моделі інтерфейсу. Почасти це відбувається через те, що основи GOMS і інших кількісних методів відомі їм настільки, що вони спочатку керуються цими методами в процесі розробки.

На практиці зазначені значення можуть варіюватися в широких межах. Для досвідченого користувача, здатного друкувати зі швидкістю 135 слів / хв, значення К може становити 0,8 с, для звичайного користувача, що має швидкість 55 слів / хв, - 0,2 с, для середнього недосвідченого користувача, що має швидкість 40 слів / хв, - 0,28 с, а для початківця - 1,2 с . Не можна сказати, що швидкість набору не залежить від того, що саме набирається. Для того щоб набрати одну букву з групи випадково взятих букв, більшості людей потрібно близько 0,5 с. Якщо ж це якийсь заплутаний код, то у більшості людей швидкість набору складе близько 0,75 символів в секунду.

Значення К включає в себе і той час, який необхідно користувачу для виправлення відразу помічених помилок.

1.11 Моделювання та тестування роботи інформаційної системи

Моделювання є достатньо ефективним способом дослідження складних систем різного призначення, – технічних, економічних, екологічних, соціальних, інформаційних – як на етапі їх проектування, так і в процесі експлуатації. Інформаційні системи розробляються творчими колективами, а однією із ключових проблем такого виробництва є проблема якості функціонування. Остання перевіряється тестовими методами. Складні системи вимагають великих обсягів засобів тестування, які потребують системного виготовлення та адекватної інтерпретації під час всього періоду експлуатації. Такі операції зручно здійснювати на основі спеціальних моделей об’єктів, що підлягають тестуванню.

Створювані інформаційні системи можуть наповнюватися моделями поведінки досліджуваних об’єктів. Узагальненням класичних аналітичних моделей є імітаційні розробки, що дозволяють прогнозувати поведінку об’єктів для якої не існує аналітичних подань.

Так, в даній роботі є клас Driver, який наслідує клас Worker, тим самим успадковує поля, функції, методи, що в подальшому полегшує роботу з програмою.

Для кращого опису об’єктів нашої системи та їхнього вивчення, використовують таке поняття як ієрархія класів – класифікація об’єктних типів, розглядаючи об’єкти як реалізацію класів і пов’язуючи їх відношеннями по типу «наслідує», «розширює», «є його абстракцією» та ін.

Тестування - важлива частина будь-якої програми контролю якості, а часто і єдина. Це сумно, тому що різноманітні методики спільної розробки дозволяють знаходити більше помилок, ніж тестування, і в той же час обходяться більш ніж удвічі дешевше в розрахунку на одну виявлену помилку. Кожен з окремих етапів тестування (блочне тестування, тестування компонентів і інтеграційне тестування) зазвичай дозволяють знайти менше 50% помилок. Комбінація етапів тестування часто призводить до виявлення менше 60% помилок.

Мета тестування протилежна цілям інших етапів розробки. Його метою є знаходження помилок. Успішним вважається тест, що порушує роботу ПЗ. Всі інші етапи розробки спрямовані на запобігання помилок та недопущення порушення роботи програми.

Тестування ніколи не доводить відсутність помилок. Відсутність помилок може вказувати як на бездоганність програми, так і на неефективність або неповноту тестів.

У моїй роботі, методи та функції я створювала поступово, щоб уникнути елементарних помилок при створенні об’єктів і їхнього опису. Цей метод допоміг заощадити час на пошук помилок. Для початку я створила лише 1 клас і працювала з ним. У ходу роботи зрозуміла, що помилок не виявлена, програма справна і можна описувати інші класи, менш громіздкі.

Після завершення написання коду, я ще декілька разів перевірила його справність і виправила помилки в інтерфейсі, для полегшення роботи користувача.

ВИСНОВКИ

Метою курсового проєкту було виконати моделювання та аналіз АІС залізничної пасажирської станції та розробити програму для виконання базових запитів для цього підприємства для кращого засвоєння і розуміння даного матеріалу.

У даній курсовій роботі було розроблено базу даних залізничної пасажирської станції. У ході роботи використовувалися класи, функції, методи, була проведена робота з пам’яттю, було розроблено алгоритм пошуку певного запиту.

Дана програма реалізовувалася у середовищі CodeBlocks з допомогою консольного меню. У ході написання коду декілька разів проводилося тестування для того, щоб переконатися у правильності роботи програми.

В результаті роботи, ми зрозуміли, що таке база даних, навчилися створювати її самостійно за допомогою мови C++

Розроблена автоматизована інформаційна система дозволяє реалізовувати управління базами даних та забезпечує відповідні функції необхідні для роботи з ними. Також, дана система дозволяє використовувати власні додаткові можливості.

Отже, С++ підтримує різні стилі програмування та надає програмісту більше можливостей зробити програму ефективною та швидкою. Сьогодні С++ широко використовується в ІТ-індустрії, в тому числі компаніями Google, Microsoft, Amazon, Facebook, Mozilla.  
На С++ пишуть 3D-ігри, складні інженерні програми, веб-сервери, криптовалюти і навіть нові мови програмування!  
Також продовжують виходити нові, сучасні стандарти мови С++.

ВИКОРИСТАНІ ДЖЕРЕЛА ІНФОРМАЦІЇ

1. Сучасні технології проектування програмного забезпечення : Вебсайт. URL: <https://sites.google.com/site/osnoviprogramnoieinzeneriie/lekciie/lekcia-6>
2. Предметна область, моделювання предметної області : Вебсайт. URL: <https://studfile.net/preview/5474325/page:2/>
3. Основні поняття об'єктно-орієнтованого програмування : Вебсайт. URL: <https://www.ua5.org/oop/384-osnovn-ponjattja-obktno-orntovanogo.html>
4. Основи програмування з використанням мови C++ : Вебсайт. URL : <https://greengofactory.jimdofree.com/it-education/>

ДОДАТКИ

Додаток А

1. Код головне меню консольної програми:

int stateM; //Стан головного меню

void MainMenu () {

cout << "Menu \n"

<< "(0) Exit the program \n"

<< "(1) Get a list and the total number of employees of the railway station \n"

<< "(2) Get a list and total number of locomotive drivers \n"

<< "(3) Get a list and total number of locomotives \n"

<< "(4) Get the list and total number of trains \n"

<< "(5) Get the list and the total number of canceled flights in full \n"

<< "(6) Get the list and total number of detained flights in full \n"

<< "(7) Get a list and total number of tickets sold \n"

<< "(8) Get a list and the total number of passengers \n"

<< "(9) Get a list and the total number of unredeemed tickets \n"

<< "(10) Get a list and the total number of returned tickets \n"

<< "Your choice \n";

cin >> stateM;

}

1. Код функції cin\_split,cin\_string:

void split\_string(string &str,ifstream &file,int size=256,char znak='-'){

str="";

getline(file,str,znak);

}

void split\_int(int &f,ifstream &file,char znak='-',int size=256){

char\* cstr= new char[size];

file.getline(cstr,size,znak);

f=atoi(cstr);

delete[] cstr;

}

1. Код класу Driver:

class Driver:public Worker{

public:

Driver( string surname,string position,int experience,string sex, int age, string children,int k\_children, int salary,string med\_review){

this->surname=surname;

this->position=position;

this->experience=experience;

this->sex=sex;

this->age=age;

this->children=children;

this->k\_children=k\_children;

this->salary=salary;

this->med\_review=med\_review;

}

void Driver\_med\_review\_true(){

if((med\_review.compare("true"))==0){

cout<<surname<<"\t "<<position<<"\t"<<experience<<"\t"<<sex<<"\t"<<age<<"\t"<<children<<"\t"<<k\_children<<"\t"<<salary<<"\t"<<med\_review<<"";

cout<<"\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

}

}

void Driver\_med\_review\_false(){

if((med\_review.compare("false"))==0){

cout<<surname<<"\t "<<position<<"\t"<<experience<<"\t"<<sex<<"\t"<<age<<"\t"<<children<<"\t"<<k\_children<<"\t"<<salary<<"\t"<<med\_review<<"";

cout<<"\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

}

}

//sex

void Driver\_sex\_male(){

if((sex.compare("male"))==0){

cout<<surname<<"\t "<<position<<"\t"<<experience<<"\t"<<sex<<"\t"<<age<<"\t"<<children<<"\t"<<k\_children<<"\t"<<salary<<"\t"<<med\_review<<"";

cout<<"\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

}

}

void Driver\_sex\_female(){

if((sex.compare("female"))==0){

cout<<surname<<"\t "<<position<<"\t"<<experience<<"\t"<<sex<<"\t"<<age<<"\t"<<children<<"\t"<<k\_children<<"\t"<<salary<<"\t"<<med\_review<<"";

cout<<"\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

}

}

//age driver

void Print\_driver\_age\_30(){

if(age==30){

cout<<surname<<"\t "<<position<<"\t"<<experience<<"\t"<<sex<<"\t"<<age<<"\t"<<children<<"\t"<<k\_children<<"\t"<<salary<<"\t"<<med\_review<<"";

cout<<"\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

}

}

void Print\_driver\_age\_40(){

if(age==40){

cout<<surname<<"\t "<<position<<"\t"<<experience<<"\t"<<sex<<"\t"<<age<<"\t"<<children<<"\t"<<k\_children<<"\t"<<salary<<"\t"<<med\_review<<""; cout<<"\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

}

}

void Print\_driver\_age\_45(){

if(age==45){

cout<<surname<<"\t "<<position<<"\t"<<experience<<"\t"<<sex<<"\t"<<age<<"\t"<<children<<"\t"<<k\_children<<"\t"<<salary<<"\t"<<med\_review<<"";

cout<<"\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

}

}

void Print\_driver\_age\_50(){

if(age==50){

cout<<surname<<"\t "<<position<<"\t"<<experience<<"\t"<<sex<<"\t"<<age<<"\t"<<children<<"\t"<<k\_children<<"\t"<<salary<<"\t"<<med\_review<<"";

cout<<"\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

}

}

// salary driver

void Print\_driver\_salary\_8000(){

if(salary==8000){

cout<<surname<<"\t "<<position<<"\t"<<experience<<"\t"<<sex<<"\t"<<age<<"\t"<<children<<"\t"<<k\_children<<"\t"<<salary<<"\t"<<med\_review<<"";

cout<<"\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

}

}

void Print\_driver\_salary\_9000(){

if(salary==9000){

cout<<surname<<"\t "<<position<<"\t"<<experience<<"\t"<<sex<<"\t"<<age<<"\t"<<children<<"\t"<<k\_children<<"\t"<<salary<<"\t"<<med\_review<<"";

cout<<"\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

}

}

void Print\_driver\_salary\_10000(){

if(salary==10000){

cout<<surname<<"\t "<<position<<"\t"<<experience<<"\t"<<sex<<"\t"<<age<<"\t"<<children<<"\t"<<k\_children<<"\t"<<salary<<"\t"<<med\_review<<"";

cout<<"\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

}

}

void Print\_driver\_salary\_12000(){

if(salary==12000){

cout<<surname<<"\t "<<position<<"\t"<<experience<<"\t"<<sex<<"\t"<<age<<"\t"<<children<<"\t"<<k\_children<<"\t"<<salary<<"\t"<<med\_review<<"";

cout<<"\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

}

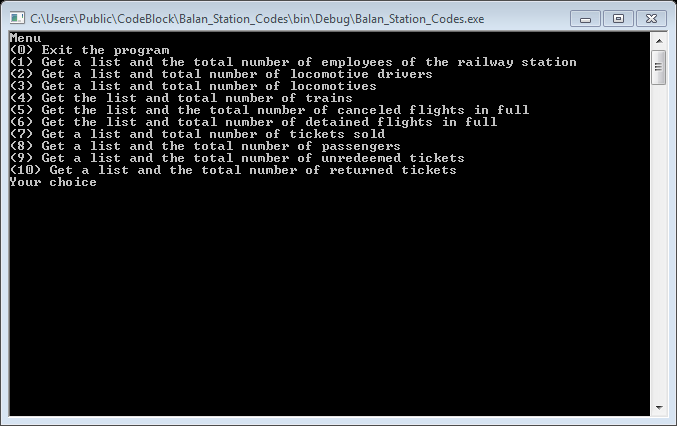
}

};

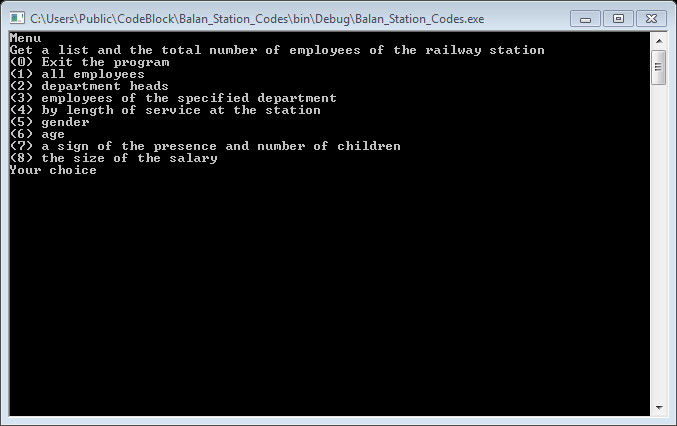
Додаток Б

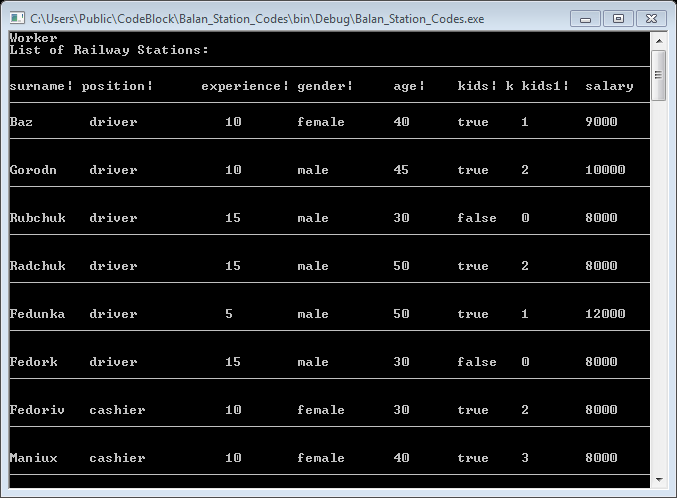
Скріншоти консольної програми.

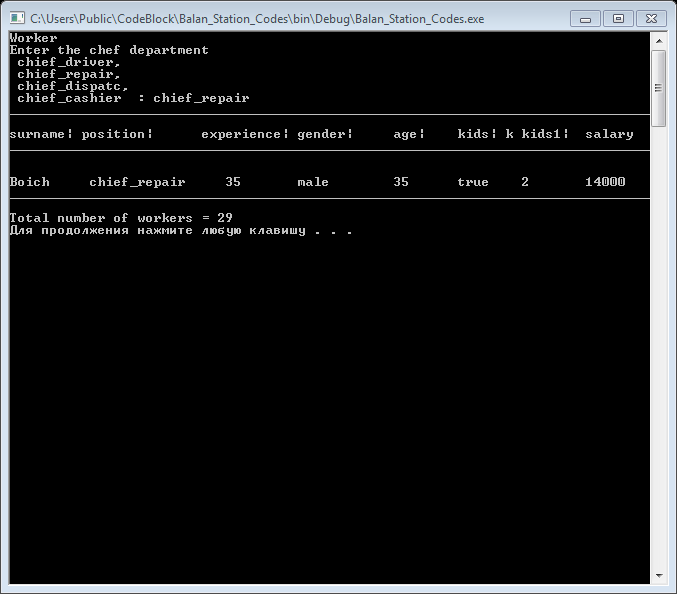
Головне меню:

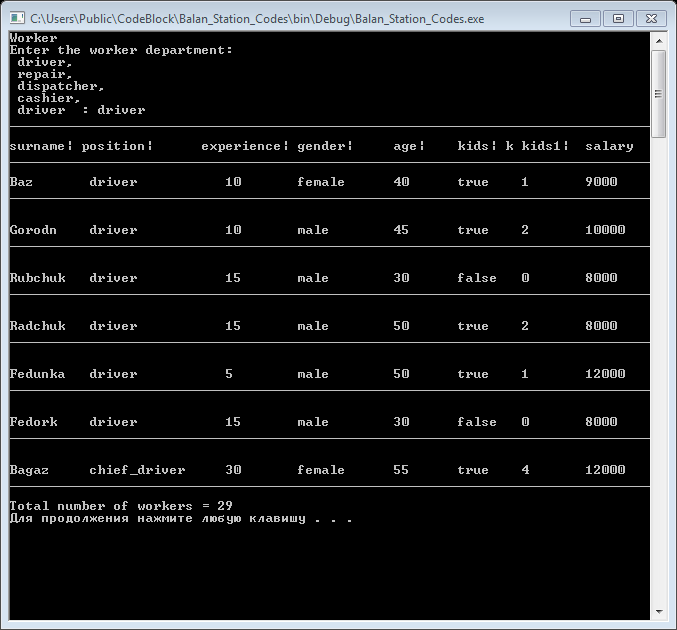


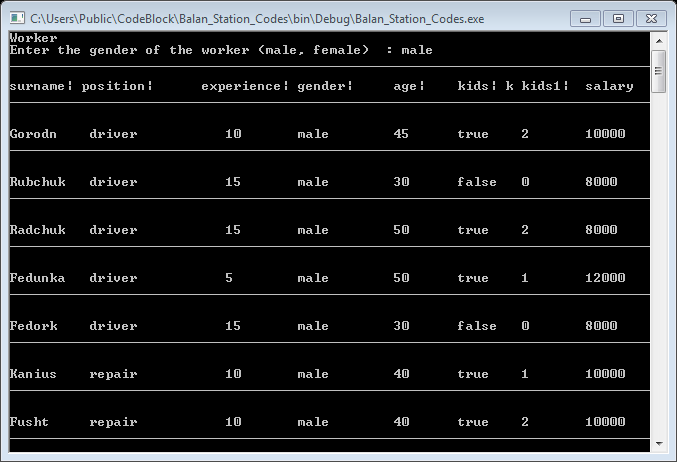
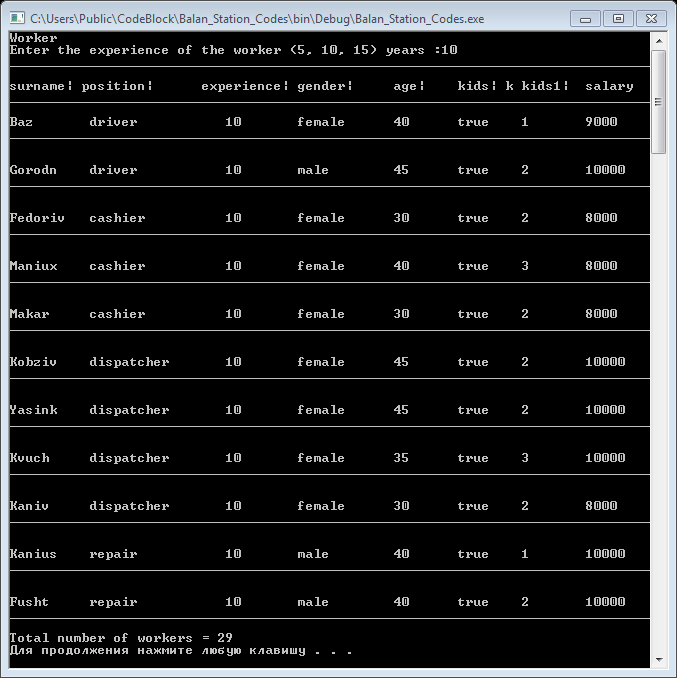
1. Запит№1

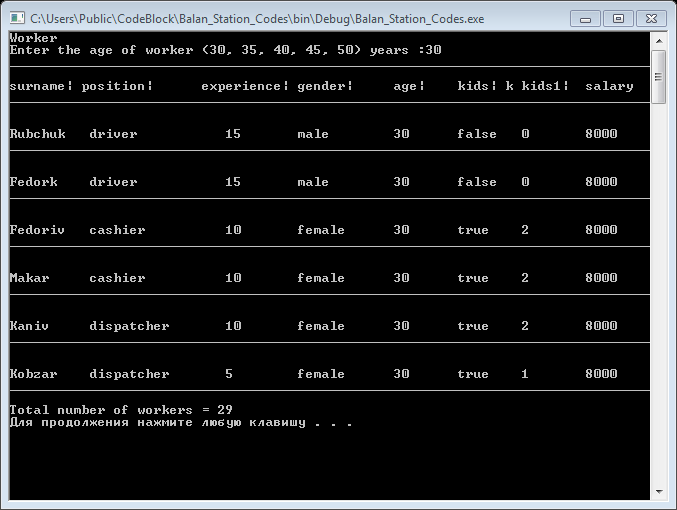


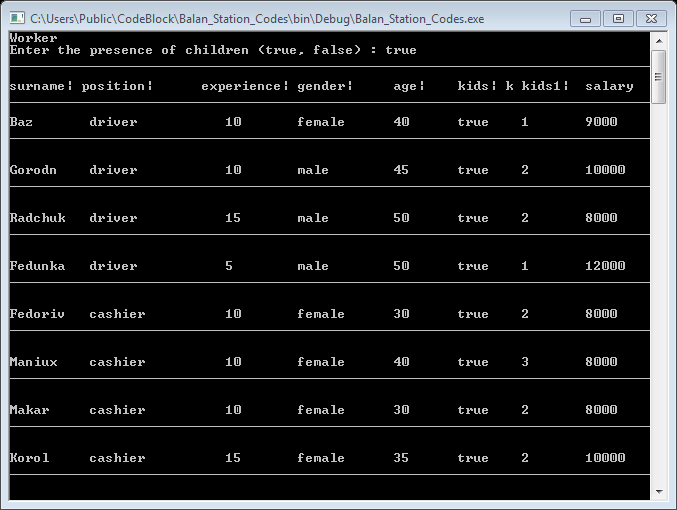


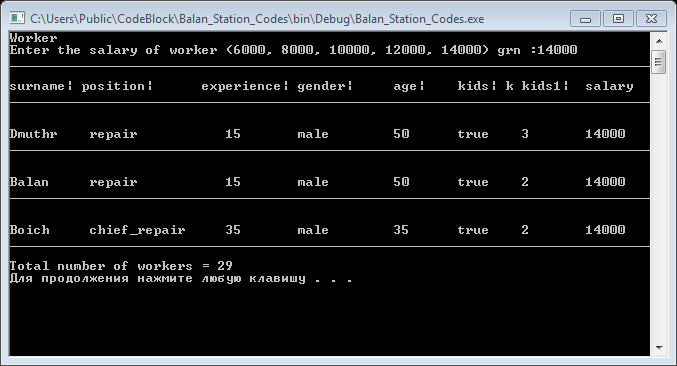




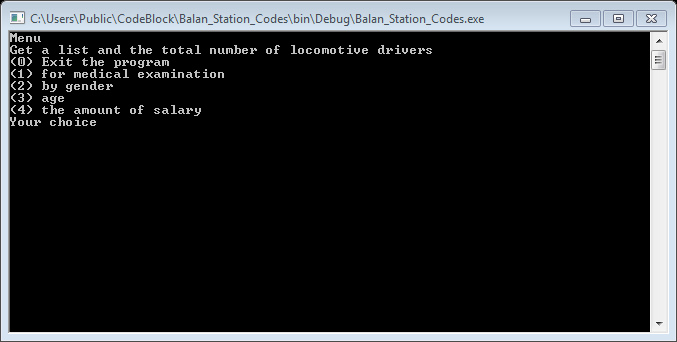


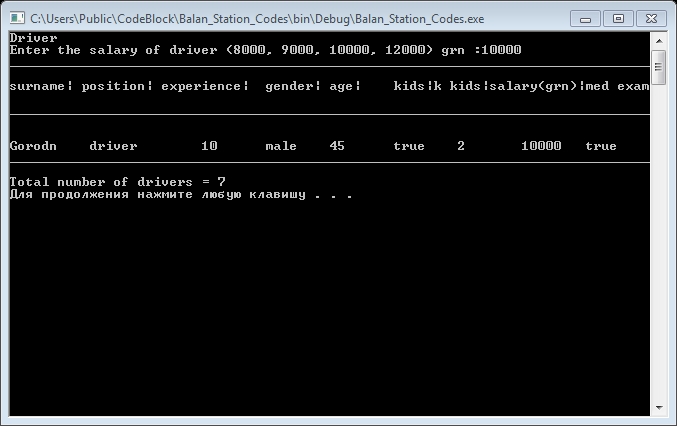




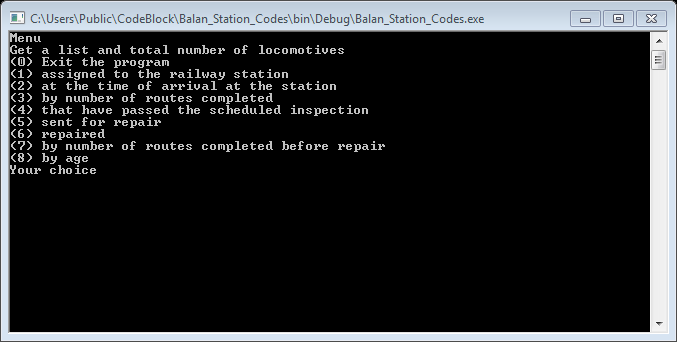


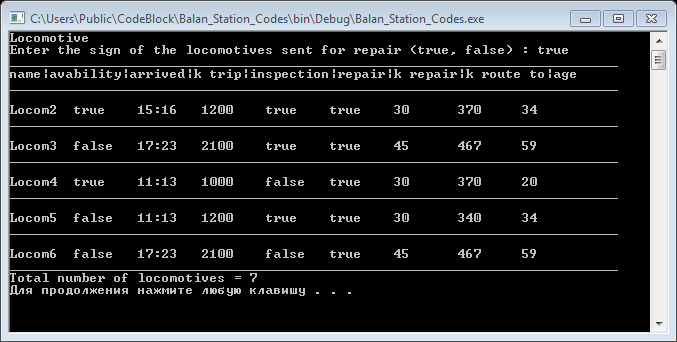
1. Запит №2



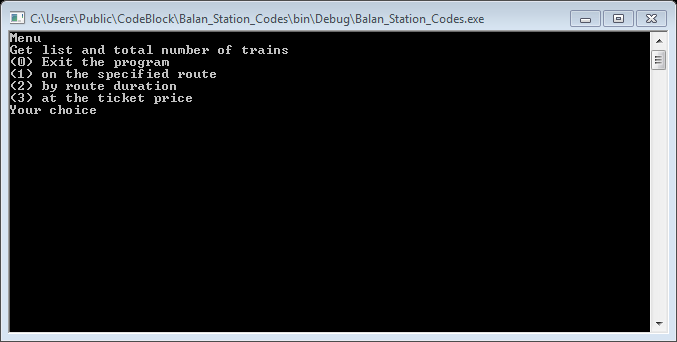


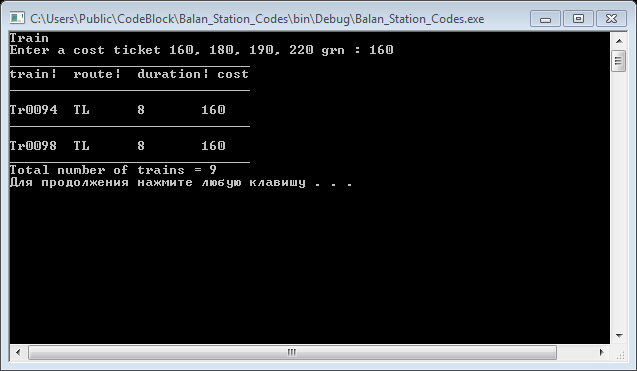
1. Запит №3



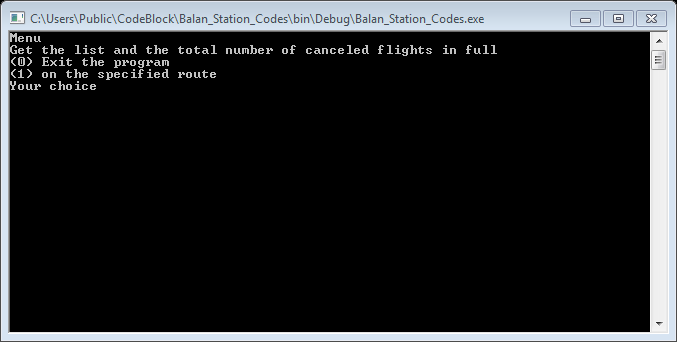


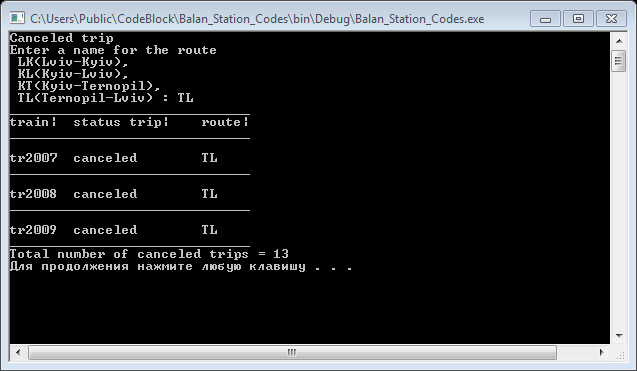
1. Запит №4



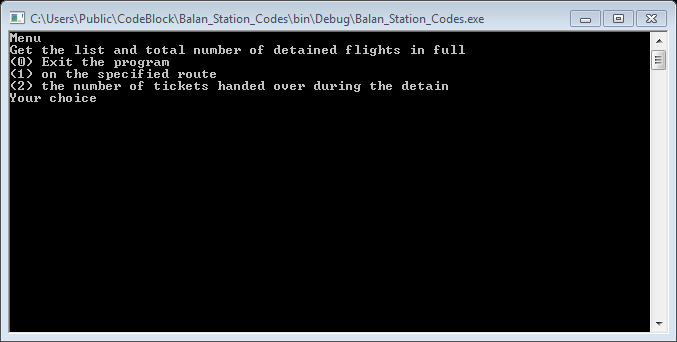


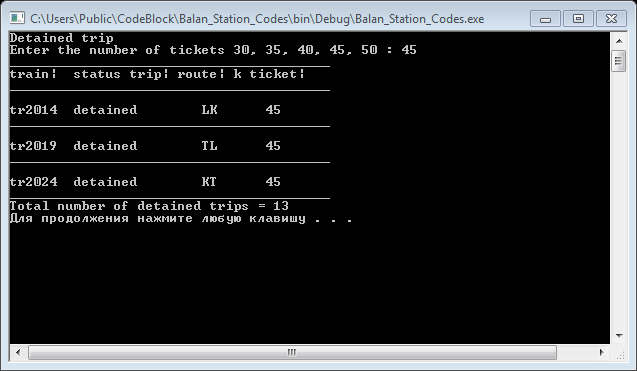
1. Запит №5



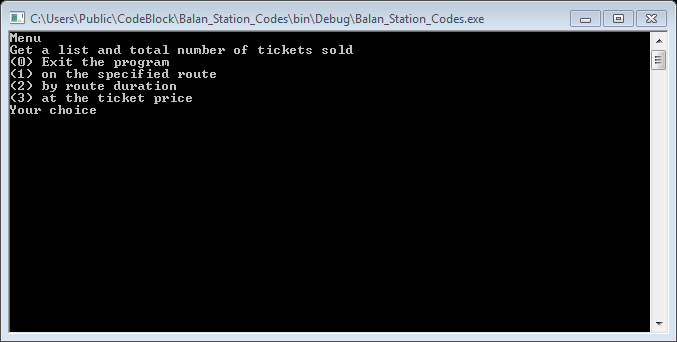


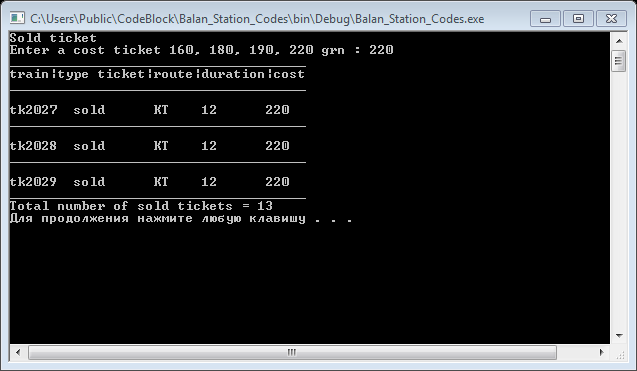
1. Запит №6



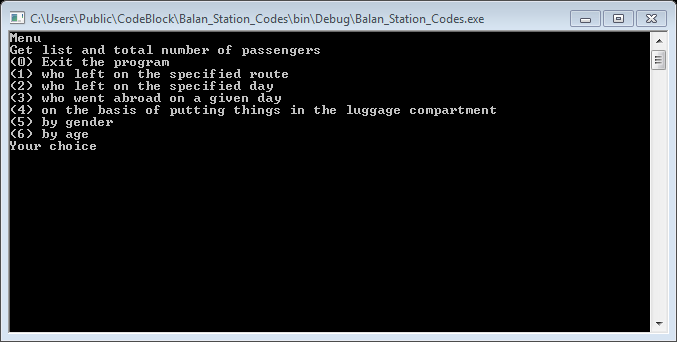


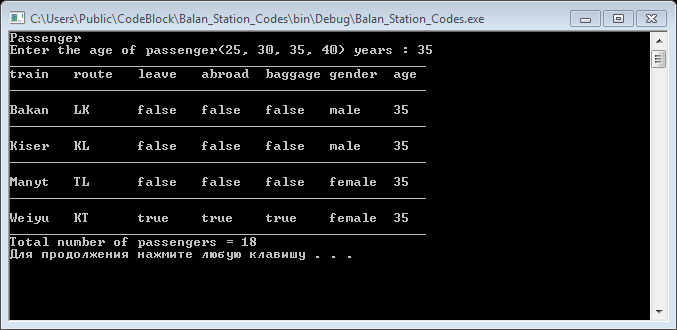
1. Запит №7



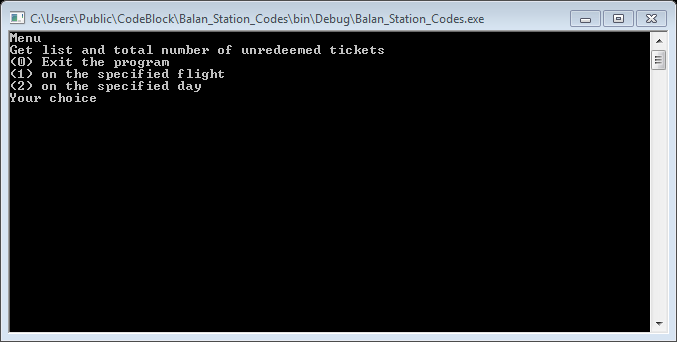


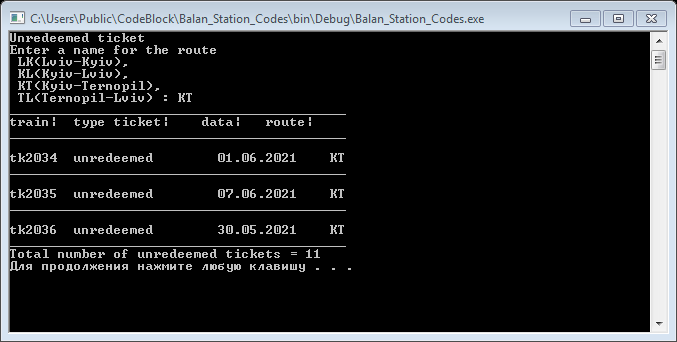
1. Запит №8





1. Запит №9





1. Запит №10

