МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ

И ПРОГРАММНОЙ ИНЖЕНЕРИИ (КАФЕДРА №43)

|  |  |  |
| --- | --- | --- |
| ОТЧЕТ ЗАЩИЩЕН С ОЦЕНКОЙ: |  |  |

ПРЕПОДАВАТЕЛЬ:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Старший преподаватель |  |  |  | Е. В. Павлов |
| (должность, уч. степень, звание) |  | (подпись, дата) |  | (инициалы, фамилия) |

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №2

«ОЦЕНКА КАЧЕСТВА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ»

ПО КУРСУ: «МЕТРОЛОГИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ»

РАБОТУ ВЫПОЛНИЛ:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| СТУДЕНТ (КА) ГР. | 4831 |  |  |  |  |  | К.А.Корнющенков |
|  |  |  | (подпись) |  | (дата отчёта) |  | (инициалы, фамилия) |

Санкт-Петербург 2020

1. Цель работы

Целью данной работы является изучение методологии оценки качества программного обеспечения.

1. Задание на лабораторную работу

Выполнить оценку качества разработанного приложения в соответствии со стандартом ГОСТ Р ИСО / МЭК 25010-2015.

Указать обоснование (пояснение) выполненных оценок качества и привести инфографику для каждой характеристики качества в виде лепестковой диаграммы.

Если количество включаемых подхарактеристик меньше трёх, то графическое представление основной характеристики приводить не нужно.

|  |
| --- |
| Вариант задания:  89. Приложение для обмена фотографиями |

1. Оценка качества при использовании

Таблица 1 — Характеристики и подхарактеристики качества при использовании

|  |  |  |  |
| --- | --- | --- | --- |
| Характеристика | | Оценка | Обоснование (пояснение) |
| 3.1 | Эффективность | 45% | Приложение по своим функциональным возможностям отвечает предполагаемым целям использования в заданном контексте работы.  С точки зрения анализа полноты и точности достижения определённых целей имеется ряд ограничений:   * Отсутствует возможность сразу просмотреть фотографию т.к в базе данных хранится ссылка на изображение; * В приложении отсутствуют изображения пользователей и из-за этого легко запутаться кто есть кто; * Сам поиск изображений и пользователей реализован крайне примитивно и, при увеличении количества изображений (пользователей ), затрудняет использование приложения в заданном контексте работы. |
| 3.2 | Производительность | 40% | В приложении не реализован контроль израсходованных ресурсов.  Все задачи управления ресурсами решаются средствами операционной системы, в рамках которой стабильная работа приложения обеспечивается экономичным форматом хранения данных и низким объёмом потребляемой памяти.  В приложении везде идет реализация vector-ов для хранения данных, а поиск реализуется линейно, что приводит к замедлению работы приложения при обработке большого количества данных. |
| 3.3 | Удовлетворённость | 45% | Оценка данной характеристики выполнена с учётом её подхарактеристик. |
| 3.3.1 | Полноценность | 20% | Приложение полностью удовлетворяет заявленным требованиям |
| 3.3.2 | Доверие | 30% | Приложение обладает достаточно простой функциональностью, выполняет свои задачи в заданном контексте работы. Но при этом все данные хранятся в текстовой файле, который никак не защищен и его легко можно изменить |
| 3.3.3 | Удовольствие | 20% | Приложение не осуществляет поддержку персональных требований пользователя и функционирует в строго заданном контексте работы в соответствии со спецификацией (сформулированными требованиями). |
| 3.3.4 | Комфорт | 10% | В приложении отсутствует персонализация настроек интерфейса, соответственно при изменении условий использования (например, уменьшение разрешения экрана, изменение освещения, etc.) эксплуатация приложения может сопровождаться физическим дискомфортом.  Не всегда по описанию можно понять, что ожидать на фотографии и из-за этого просмотри может вызывать дискомфорт т.к каждую фотографию надо искать дополнительно по ссылку в браузере. |
| 3.4 | Свобода от риска | 15% | Оценка данной характеристики выполнена с учётом её подхарактеристик. |
| 3.4.1 | Смягчение отрицательных последствий экономического риска | 10% | Приложение не осуществляет смягчение последствий экономических рисков для пользователей, в приложении отсутствует какая-либо информация о мерах предосторожности. |
| 3.4.2 | Смягчение отрицательных последствий риска для здоровья и безопасности | 25% | Приложение не осуществляет смягчение последствий рисков, связанных со здоровьем и безопасностью пользователей, например, в случае недостоверного описания нельзя будет сразу понять, что изображено на фотографии. |
| 3.4.3 | Смягчение отрицательных последствий экологического риска | 10% | Контекст работы приложения не предусматривает возможные экологические риски, а также их снижение. |
| 3.5 | Покрытие контекста | 40% | Оценка данной характеристики выполнена с учётом её подхарактеристик. |
| 3.5.1 | Полнота контекста | 60% | Приложение не поддерживает настройку или персонализацию интерфейса, что в определённых условиях может затруднить его использование (например, при низком разрешении экрана).  Актуальность данных приложения зависит от процессов, которые выходят за рамки его функционирования, следовательно, использование приложения в контексте актуальности предоставляемых им данных может быть сопряжено с определёнными рисками для пользователя (данное положение продиктовано сформулированной концепцией продукта и не должно значительно снижать оценку). |
| 3.5.2 | Гибкость | 20% | Приложение не предполагает использование за рамками его спецификации и не предусматривает какую-либо адаптацию под другие задачи или группы пользователей; в тоже время формат хранения данных допускают возможность изменения для адаптации к другим категориям пользователей. |

1. Оценка качества продукта

Таблица 2 — Модель качества продукта

|  |  |  |  |
| --- | --- | --- | --- |
| Характеристика | | Оценка | Обоснование (пояснение) |
| 4.1 | Функциональная пригодность | 63% | Оценка данной характеристики выполнена с учётом её подхарактеристик. |
| 4.1.1 | Функциональная полнота | 65% | Приложение в целом отвечает всем определённым задачам и целям пользователя в указанных условиях использования, однако отсутствие рекомендательной системы и подсказок при поиске снижает данную оценку. |
| 4.1.2 | Функциональная корректность | 95% | В целом все операции ввода-вывода соответствуют ожидаемому результату. |
| 4.1.3 | Функциональная целесообразность | 30% | В приложении плохая система навигации, нельзя сделать определенные действия, который подразумеваются по логике. |
| 4.2 | Уровень производительности | 52% | Оценка данной характеристики выполнена с учётом её подхарактеристик. |
| 4.2.1 | Временные характеристики | 45% | Функциональные возможности и архитектура приложения допускают уменьшение производительности при любых условиях использования. |
| 4.2.2 | Использование ресурсов | 60% | Приложение неэффективно использует человеческие ресурсы (время) в условиях увеличения количества вакансий. |
| 4.2.3 | Потенциальные возможности | 50% | Устойчивость работы приложения зависит от достижения предельных значений параметров, однако подобные условия работы негативно повлияют на функциональную пригодность приложения. |
| 4.3 | Совместимость | 87,5% | Оценка данной характеристики выполнена с учётом её подхарактеристик. |
| 4.3.1 | Сосуществование | 95% | Приложение не влияет на работу других приложений или систем. |
| 4.3.2 | Функциональная совместимость | 80% | Для хранения информации приложение использует текстовые файлы, что обеспечивает возможность стороннего доступа к данным и совместное их использование. Иные способы для взаимодействия и функционирования с другими приложениями не предусмотрены.  Возможности совместимости с другими операционными системами не рассматриваются. |
| 4.4 | Удобство использования | 44% | Оценка данной характеристики выполнена с учётом её подхарактеристик. |
| 4.4.1 | Определение пригодности | 80% | С точки зрения пользователя приложение в достаточной мере согласуется с функциональной целесообразностью: все наименования элементов меню приняты согласно контексту работы приложения, а также присутствует информация о назначении продукта. |
| 4.4.2 | Изучаемость | 95% | Простая функциональность и удобная навигация исключают необходимость в обучении. |
| 4.4.3 | Управляемость | 35% | Приложение неудобно в навигации, некоторые шаги не всегда очевидны, при вводе некорректных данных нельзя отменить действие |
| 4.4.4 | Защищённость от ошибки пользователя | 30% | В приложении почти отсутствуют проверки данных на корректность, грамматику и другие факторы, что может впоследствии затруднить работу других пользователей |
| 4.4.5 | Эстетика пользова-тельского интерфейса | 20% | Учитывая назначение приложения, наличие консольного интерфейса выступает фактором, который в значительной степени снижает оценку данной характеристики. |
| 4.4.6 | Доступность | 5% | Приложение не осуществляет поддержку категорий пользователей с ограниченными возможностями, также отсутствует возможность настройки и персонализации интерфейса. |
| 4.5 | Надежность | 95% | Оценка данной характеристики выполнена с учётом её подхарактеристик. |
| 4.5.1 | Завершенность | 95% | Критерии, которые могли бы негативно сказаться на надёжности приложения в условиях нормальной работы, не выявлены. |
| 4.5.2 | Готовность | 95% | Приблизительная (субъективная) оценка способности приложения находиться в рабочем состоянии, составляет ~99,9%.  Возможные отказы приложения могут быть связаны исключительно с внешними факторами. |
| 4.5.3 | Отказоустойчивость | 95% | По результатам тестирования функциональных требований, в приложении не было выявлено критических дефектов, что в сочетании с анализом архитектуры приложения позволяет произвести высокую оценку данного атрибута качества. |
| 4.5.4 | Восстанавливаемость | 95% | Архитектура приложения исключает влияние сбоев на целостность и корректность данных.  Восстановление приложения обеспечивается перезапуском программы и не зависит от причин, которые вызвали отказ. |
| 4.6 | Защищенность | 5% | Оценка данной характеристики выполнена с учётом её подхарактеристик. |
| 4.6.1 | Конфиденциальность | 5% | В приложении не реализовано разграничение прав доступа. |
| 4.6.2 | Целостность | 5% | Нет защиты или контроля целостности и корректности данных, при этом возможно изменение хранимой информации с помощью сторонних приложений, так как все данные приложения хранятся во внешнем файле. |
| 4.6.3 | Неподдельность | 5% | Приложение не поддерживает функциональность, которая бы позволила обеспечить контроль данных и производимых операций над ними. |
| 4.7 | Сопровождаемость, модифицируемость | 73% | Оценка данной характеристики выполнена с учётом её подхарактеристик. |
| 4.7.1 | Модульность | 70% | Присутствует разделение пользовательского интерфейса и данных приложения на отдельные составляющие. В данном случае пользовательский интерфейс рассматривается как совокупность уровня представления (отображение данных пользователю) и контроллера (управляющей логики приложения). Хранение данных приложения осуществляется посредством внешних файлов, таким образом, изменение данных может осуществляться независимо от пользовательского интерфейса.  Реализации модульной структуры в приложении не предусмотрена. |
| 4.7.2 | Возможность многократного использования | 50% | Реализация приложения представляет собой простой и самодокументируемый код, который писался в формате mvp(Минимально жизнеспособный продукт) из-за чего присутствуют костыли |
| 4.7.3 | Анализируемость | 80% | В приложении не реализованы возможности генерации отчётов об ошибках или логов функционирования, однако анализ и диагностика данного приложения не сопряжены с какими-либо трудностями вследствие его простоты. |
| 4.7.4 | Модифицируемость | 70% | Приложение представляет собой один программный модуль с достаточно простой функциональностью и низкими требованиями к сопровождению. |
| 4.7.5 | Тестируемость | 95% | Функциональные возможности приложения обуславливают простоту его тестирования, как в отношении требований, так и в отношении кода. |
| 4.8 | Переносимость, мобильность | 80% | Оценка данной характеристики выполнена с учётом её подхарактеристик. |
| 4.8.1 | Адаптируемость | 85% | Приложение не требует существенных изменений исходного кода для портирования в иную среду функционирования. |
| 4.8.2 | Устанавливаемость | 60% | Приложение не требует установки, однако функциональность приложения обеспечивается наличием соответствующих библиотек. |
| 4.8.3 | Взаимозаменяемость | 95% | Обновление версии приложения не влияет на используемые форматы хранения данных. |

Выводы по работе

В результате выполнения данной работы произведена оценка разработанного приложения с точки зрения моделей качества при использовании и качества продукта в соответствии со стандартом ГОСТ Р ИСО / МЭК 25010-2015.

Для всех оценок характеристик качества приложения приведено обоснование или формальное уточнение.

Ниже представлены оценки характеристик моделей качества, в том числе в графическом виде для полноты информации (что может быть важно при отсутствии консолидированных метрик качества).

*Оценка качества при использовании:*

1. Эффективность — 45%
2. Производительность — 40%
3. Удовлетворённость — 45%
4. Свобода от риска — 15%

В приложении не реализованы механизмы для снижения различного рода рисков, поэтому для данной характеристики инфографика не приводится.

1. Покрытие контекста — 40%

*Оценка качества продукта:*

1. Функциональная пригодность — 63%
2. Уровень производительности — 52%
3. Совместимость — 87,5%
4. Удобство использования — 44%
5. Надежность — 95%

Приложение обладает достаточно простой архитектурой, поэтому для данной характеристики инфографика не приводится.

1. Защищенность — 5%

Приложение не поддерживает защиту информации и данных, поэтому для данной характеристики инфографика не приводится.

1. Сопровождаемость, модифицируемость — 73%
2. Переносимость, мобильность — 80%

Таким образом, по результатам проделанной работы можно сделать выводы о сильных и слабых сторонах продукта. Отдельно следует отметить относительно низкие оценки удовлетворённости, ащищенность, функциональной пригодности и удобства использования, что обусловлено главным образом следующими факторами:

* Отсутствием полной информации о фотографиях;
* Отсутствием критериев или фильтров для поиска;
* Реализация консольного интерфейса для задач, где он менее всего для этого подходит;
* Отсутствие защищенности данных

Указанные факторы представляют собой основные недостатки приложения и в значительной степени оказали отрицательное влияние на оценку некоторых его характеристик качества.

Стоит отметить, что выполненные оценки в значительной степени несут субъективный характер, так как рассматриваемый стандарт не предполагает использование точных численных методов для оценки качества. Соответственно, полученные оценки характеристик качества могут отличаться от результатов подобной работы, выполненной иными лицами, по отношению к данному приложению.

Таким образом, можно заключить, что выполненная работа соответствует поставленной задаче и отвечает всем сформулированным в методических указаниях требованиям.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. ГОСТ Р ИСО/МЭК 25010-2015 Информационные технологии. Системная и программная инженерия. Требования и оценка качества систем и программного обеспечения (SQuaRE). Модели качества систем и программных продуктов, 2015.
2. ГОСТ Р ИСО/МЭК 25051-2017 Информационные технологии (ИТ). Системная и программная инженерия. Требования и оценка качества систем и программного обеспечения (SQuaRE). Требования к качеству готового к использованию программного продукта (RUSP) и инструкции по тестированию, 2017.
3. Липаев В. В. Выбор и оценивание характеристик качества программных средств. Методы и стандарты / В. В. Липаев. — М.: СИНТЕГ, 2001. — 228 с.: ил.
4. Липаев В. В. Проектирование и производство сложных заказных программных продуктов / В. В. Липаев. — М.: СИНТЕГ, 2011. — 408 с.

ПРИЛОЖЕНИЕ А

Исходный код программы

#include <iostream>

#include <fstream>

#include <vector>

#include <string>

using namespace std;

//MARK: Cтруктуры

struct Photo{

string href;

string description;

};

struct User{

string login;

string password;

vector<string> followers;

vector<Photo> photoVector;

};

//MARK: Class

class Instagramm{

private:

const string path = "/Users/kirill/Desktop/tinder/tinder/file.txt";

vector<User> dataAllUser;

User activeUser;

//обработка строки из файла

void readDataSupport(string data);

// регистрация

void registration();

// вход

void entrance();

//menu

void menu();

//add image

void addImage();

//finc people

void findPeople();

//просмотр фото

void lookPhotoActiveUser();

//просмотр подписчиков

void lookFollowers();

//редактирование данных своих

void editData();

//редактирование описания у фотографий

void editDescriptionPhoto();

//удаление фотографий

void removePhoto();

//save

void saveAndClose();

//старт

void start();

//info

void info();

public:

//считывание данных

void readData();

};

int main() {

Instagramm insta;

insta.readData();

return 0;

}

//MARK: Считывание данных из файла

void Instagramm::readDataSupport(string data){

User user;

string line;

int count = 0;

vector<string> followers;

vector<Photo> photoVector;

Photo photo;

for (int i=0;i<data.length();i++){

if (data[i]!='#' && count==0){

line+=data[i];

}else if (data[i]=='#' && count==0){

user.login = line;

line = "";

count++;

}else if (data[i]!='#' && count==1){

line+=data[i];

}else if (data[i]=='#' && count==1){

user.password = line;

count++;

line = "";

}else if (data[i]!='\*' && count==2){

if (data[i]!='!'){

line+=data[i];

}else{

followers.push\_back(line);

line = "";

}

}else if (data[i]=='\*' && count==2){

count++;

user.followers = followers;

line = "";

}else{

if (data[i]!=' ' && data[i]!='|'){

line+=data[i];

}else if (data[i]==' '){

photo.description = line;

line = "";

}else if (data[i]=='|'){

photo.href = line;

line = "";

photoVector.push\_back(photo);

}

}

}

user.photoVector = photoVector;

dataAllUser.push\_back(user);

}

void Instagramm::readData(){

ifstream file(path);

if (!file) {

cout << "файл не найден" << endl;

} else {

string data;

while (getline(file, data)) {

readDataSupport(data);

}

file.close();

}

start();

}

//MARK: Вход в приложение

void Instagramm::start(){

int choose;

cout << "Выберите пункт меню " << endl;

cout << "1 - авторизация " << endl;

cout << "2 - регистрация " << endl;

cin >> choose;

switch (choose) {

case 1:

entrance();

break;

case 2:

registration();

break;

default:

cout << "error" << endl;

break;

}

}

void Instagramm::registration(){

string login, password, password2;

cout << "Введите логин " << endl;

cin >> login;

cout << "Введите пароль " << endl;

cin >> password;

cout << "Введите пароль еще раз " << endl;

cin >> password2;

if (password == password2){

cout << "Вы успешно зарегистрировались " << endl;

User user;

user.login = login;

user.password = password;

dataAllUser.push\_back(user);

activeUser = user;

menu();

}else{

cout << "Пароли не совпадают " << endl;

}

}

void Instagramm::entrance(){

string login, password;

cout << "Введите логин " << endl;

cin >> login;

cout << "Введите пароль " << endl;

cin >> password;

for(int i=0;i<dataAllUser.size();i++){

if (dataAllUser[i].login == login){

if(dataAllUser[i].password == password){

activeUser = dataAllUser[i];

cout << "Вы успешно авторизировались " << endl;

menu();

}else {

cout << "Пароль введен неверно " << endl;

}

break;

}

}

}

//MARK: меню

void Instagramm::menu(){

int choose = -1;

system("cls");

while (choose!=8) {

cout << "Меню" << endl;

cout << "1 - просмотр своих фотографий" << endl;

cout << "2 - добавление фотографий " << endl;

cout << "3 - удаление своих фотографий " << endl;

cout << "4 - редактирование описания у фотографий " << endl;

cout << "5 - редактирование своих данный login/password " << endl;

cout << "6 - просмотр подписок " << endl;

cout << "7 - поиск пользователей " << endl;

cout << "8 - выход и сохранение данных " << endl;

cout << "9 - информация о программе " << endl;

cin >> choose;

switch (choose) {

case 1:

lookPhotoActiveUser();

break;

case 2:

addImage();

break;

case 3:

removePhoto();

break;

case 4:

editDescriptionPhoto();

break;

case 5:

editData();

break;

case 6:

lookFollowers();

break;

case 7:

findPeople();

break;

case 8:

saveAndClose();

break;

case 9:

info();

break;

default:

break;

}

}

}

void Instagramm::addImage(){

Photo photo;

cout << "Введите ссылку на фотографию " << endl;

cin >> photo.href;

cout << "Введите описание " << endl;

cin >> photo.description;

activeUser.photoVector.push\_back(photo);

cout << "фото успешно добавлено " << endl;

}

void Instagramm::findPeople(){

string login;

bool find = false;

cout << "Введите логин пользователя " << endl;

cin >> login;

for (int i=0;i<dataAllUser.size();i++){

if(dataAllUser[i].login == login){

find = true;

cout << "Пользователь найден " << endl;

cout << "1 - подписаться" << endl;

cout << "2 - просмотреть фотографии " << endl;

int choose;

cin >> choose;

switch (choose) {

case 1:

activeUser.followers.push\_back(dataAllUser[i].login);

cout << "Вы успешно подписались" << endl;

break;

case 2:

for (int j=0;i<dataAllUser[j].photoVector.size();j++){

cout << dataAllUser[i].photoVector[j].href << " " << dataAllUser[i].photoVector[j].description << endl;

}

if (dataAllUser[i].photoVector.size() == 0){

cout << "у пользователя нету фото" << endl;

}

break;

default:

break;

}

break;

}

}

if (!find){

cout << "Пользователь не найден " << endl;

}

}

void Instagramm::lookPhotoActiveUser(){

for (int i=0;i<activeUser.photoVector.size();i++){

cout << activeUser.photoVector[i].description << "->" << activeUser.photoVector[i].href << endl;

}

if (activeUser.photoVector.size() == 0){

cout << "у вас нету фото" << endl;

}

}

void Instagramm::lookFollowers(){

for (int i=0;i<activeUser.followers.size();i++){

cout << activeUser.followers[i] << endl;

}

if (activeUser.followers.size() == 0){

cout << "у вас нету подписчиков" << endl;

}

}

void Instagramm::editData(){

cout << "Что вы хотите изменить ?" << endl;

cout << "1 - логин" << endl;

cout << "2 - пароль" << endl;

int choose;

cin >> choose;

switch (choose) {

case 1:{

string login;

int index = 0;

cout << "Введите новый логин" << endl;

cin >> login;

for (int i=0;i<dataAllUser.size();i++){

if (dataAllUser[i].login == activeUser.login){

index = i;

break;

}

}

activeUser.login = login;

dataAllUser[index] = activeUser;

cout << "логин успешно изменен" << endl;

break;

}

case 2:{

string password,password2;

cout << "Введите новый пароль " << endl;

cin >> password;

cout << "Повторите новый пароль" << endl;

cin >> password2;

if (password == password2){

activeUser.password = password;

cout << "Пароль успешно изменен" << endl;

}else{

cout << "Пароли разные " << endl;

}

break;

}

default:

cout << "error" << endl;

break;

}

}

void Instagramm::editDescriptionPhoto(){

cout << "Выберите номер фотографии, описание которой хотите изменить" << endl;

for (int i=0;i<activeUser.photoVector.size();i++){

cout << i+1 << ":" << activeUser.photoVector[i].description << " " << activeUser.photoVector[i].href << endl;

}

int index;

cin >> index;

cout << "Вы выбрали изображение :" << endl;

cout << activeUser.photoVector[index-1].description << " " << activeUser.photoVector[index-1].href << endl;

string description;

cout << "Введите новое описание" << endl;

cin >> description;

activeUser.photoVector[index-1].description = description;

cout << "Описание успешно изменёно" << endl;

}

void Instagramm::removePhoto(){

cout << "Выберите номер фотографии, которую хотите удалить " << endl;

for (int i=0;i<activeUser.photoVector.size();i++){

cout << i+1 << ":" << activeUser.photoVector[i].description << ":" << activeUser.photoVector[i].href << endl;

}

int index;

cin >> index;

cout << "Вы выбрали изображение :" << endl;

cout << activeUser.photoVector[index-1].href << " " << activeUser.photoVector[index-1].description << endl;

activeUser.photoVector.erase(activeUser.photoVector.begin()+index-1);

cout << "Изображение успешно удалено"<< endl;

}

void Instagramm::saveAndClose(){

ofstream file(path);

if (file.is\_open()){

for (int i=0;i<dataAllUser.size();i++){

if (dataAllUser[i].login == activeUser.login){

dataAllUser[i] = activeUser;

break;

}

}

for (int i=0;i<dataAllUser.size();i++){

file << dataAllUser[i].login << "#" << dataAllUser[i].password << "#";

for (int j=0;j<dataAllUser[i].followers.size();j++){

file << dataAllUser[i].followers[j] << "!";

}

file << "\*";

for (int k=0;k<dataAllUser[i].photoVector.size();k++){

file << dataAllUser[i].photoVector[k].description << " " << dataAllUser[i].photoVector[k].href << "|";

}

file << endl;

}

}else{

cout << "файл не найден " << endl;

}

}

void Instagramm::info(){

cout << "Приложение для обмена фотографиями" << endl;

cout << "Разработчик - Кирилл Корнющенков" << endl;

cout << "Год выпуска - 2020" << endl;

}