МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ 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 |
| «ПОСТРОЕНИЕ МОДЕЛИ КАЧЕСТВА ПРИ ИСПОЛЬЗОВАНИИ» |
| по дисциплине: «МЕТРОЛОГИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ» |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. | 4631 |  |  |  | С.А. Гришин |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург   
2018

1. **Цель работы**

Целью данной работы является изучение оценочных показателей качества и получение навыков описания модели качества программного обеспечения.

1. **Задание**

Описать модель качества программного продукта при использовании для программы из ЛР №1 в соответствии со стандартом ГОСТ Р ИСО/МЭК 25010-2015.

1. **Модель качества при использовании**

Таблица 3.1 — Характеристики и подхарактеристики качества при использовании.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Характеристика** | **Оценка** | **Пояснение** |
| 1. | Эффективность | Высокая | Приложение в полной мере отвечает критериям точности и полноте получаемых данных результатов при использовании. |
| 2. | Производительность | Средняя | Приложение не предполагает контроль израсходованных ресурсов, например, при недостаточном количестве памяти программа прекратит свою работу с ошибкой. |
| 3. | Удовлетворенность | Высокая | Приложение отвечает всем требованиям в контексте использования. |
| 3.1 | Полноценность | Высокая | Функциональность программы в полной мере соответствует поставленной задаче. |
| 3.2 | Доверие | Высокая | Обрабатываемые данные программой, полностью прозрачны для пользователя в любой момент. |
| 3.3 | Удовольствие | Высокая | Пользователь может просмотреть внутреннее состояние таблицы, а так же меня её содержимое |
| 3.4 | Комфорт | Высокая | Программа работает одним потоком, отсутствует обработка нескольких подключений. |
| 4. | Свобода от риска | Высокая | В приложении отсутсвуют риски. |
| 4.1 | Смягчение отрицательных последствий экономического риска | Высокая | В приложении отсутсвуют риски. |
| 4.2 | Смягчение отрицательных последствий риска здоровья и безопасности | Высокая | В приложении отсутсвуют риски. |
| 4.3 | Смягчение отрицательных последствий экологического риска | Высокая | В приложении отсутсвуют риски. |
| 5. | Покрытие контекста | Среднее | Выход за спецификации использования не предусмотрен. |
| 5.1 | Полнота контекста | Высокое | Приложение в полной мере выполняет свои функции в рамках указанных условий использования. |
| 5.2 | Гибкость | Средняя | При необходимости приложение может быть расширено за рамки изначально предусмотренных спецификаций использования. |

Таблица 2.6.2 — Модель качества продукта

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Характеристика** | **Оценка** | **Пояснение** |
| 1. | Функциональная  пригодность | Высокая | Приложение в полной мере отвечает всем заявленным и предполагаемым требованиям. |
| 1.1 | Функциональная полнота | Высокая | Приложение охватывает все цели и задачи пользователя при использовании (в рамках контекста). |
| 1.2 | Функциональная  корректность | Высокая | Приложение обеспечивает необходимую степень точности результатов использования. |
| 1.3 | Функциональная  целесообразность | Высокая | Приложение выполняет все необходимые действия вместо клиента. |
| 2. | Уровень  производительности | Высокая | Приложение имеет достаточную производительность, для отображения информации пользователю. |
| 2.1 | Временные  характеристики | Высокая | При высокой нагрузке производительность снижается в разы. |
| 2.2 | Использование ресурсов | Высокая | Программа работает только для одного пользователя и из-за этого не требовательна к ресурсам. |
| 2.3 | Потенциальные  возможности | Среднее | Стабильная работа приложения гарантируется при встраивании программы в качестве отдельного модуля корректно работает при установленных размерах данных. |
| 3. | Совместимость | Высокая | У приложения присутствует (API) интерфейс обмена данными с другими приложениями. |
| 3.1 | Сосуществование | Высокая | Приложение незначительно влияет на другие процессы. |
| 3.2 | Интероперабельность | Отсутствует | Приложение не имеет возможности обмениваться и использовать информацию от других приложений/систем/компонентов. |
| 4. | Удобство использования | Среднее | Минималистичный интерфейс позволяет пользователю установить контакт с приложением. |
| 4.1 | Определимость  пригодности | Среднее | Неподготовленный пользователь с трудом поймет назначение приложения и область его применения. |
| 4.2 | Изучаемость | Среднее | Для использование не обязательно полностью изучать работу, так же имеется файл с описанием используемых алгоритмов. |
| 4.3 | Управляемость | Среднее | Пользователь может добавлять и удалять информацию из таблицы, а так же выводить её на экран. |
| 4.4 | Защищенность от ошибки пользователя | Высокая | При неверных параметрах приложение проигнорирует их или сообщит об этом. |
| 4.5 | Эстетика  пользовательского  интерфейса | Среднее | Интерфейс реализован с помощью командной строки. |
| 4.6 | Доступность | Высокая | Приложение работает в с возможностью работать с ним только одном пользователю. |
| 5. | Надежность | Среднее | Приложение стабильно работает в |
| 5.1 | Завершенность | Высокая | Приложение полностью соответствует требованиям надежности при нормальной работе. |
| 5.2 | Готовность | Высокая | При использовании приложения не происходит прерывания программы. |
| 5.3 | Отказоустойчивость | Высокая | При нарушении работы сервера и/или отсутствия необходимого запаса ОЗУ может произойти «падение» приложения. |
| 5.4 | Восстанавливаемость | Низкая | Восстановить данные, с которыми приложение работало в момент отказа, невозможно. |
| 6. | Защищенность | Высокая | Все пользователи приложения имеют одинаковые права. |
| 6.1 | Конфиденциальность | Высокая | Приложение не запрашивает конфиденциальных данных от пользователя |
| 6.2 | Целостность | Высокая | Приложение не имеет доступа к модификации данных в памяти. |
| 7. | Сопровождаемость | Высокая | Приложение разбито на отдельные функции/методы, выполняющие поставленные задачи, которые понятны по контексту и прилагаемым файлам к приложению. |
| 7.1 | Модульность | Высокая | Приложение зависит только от стандартной библиотеки и может использоваться как модуль. |
| 7.2 | Возможность  многократного  использования | Высокая | Данное приложение можно использовать как модуль в другом приложении или системе. |
| 7.3 | Анализируемость | Низкая | Отсутствует лог, покрытие тестами не выполнено. |
| 7.4 | Модифицируемость | Высокая | Приложению требуется только стандартная библиотека. |
| 7.5 | Тестируемость | Средняя | Тестирование приложения является в достаточной степени трудоемкой задачей ввиду технических особенностей и контекста использования. |
| 8. | Переносимость | Высокая | Приложение может работать в любой ОС, для которых есть компилятор C++ |
| 8.1 | Адаптируемость | Высокая | Приложение не требует изменений при переносе в другую среду. |
| 8.2 | Устанавливаемость | Высокая | Установка приложения не требуется. На выходе исполняемый файл. |
| 8.3 | Взаимозаменяемость | Высокая | Существует множество аналогов данного приложения. |

1. **Выводы**

В результате выполнения данной лабораторной работы были получены навыки описания модели качества программного продукта при использовании в соответствии со стандартом ГОСТ Р ИСО/МЭК 25010-2015.

Составлена модель качества программного продукта при использовании, включающая в себя пять характеристик: эффективность, производительность, удовлетворенность, свободу от риска и покрытие контекста.

В ходе выполнения ЛР некоторые характеристики (или под характеристики) не были корректно оценены в виду отсутствия полной информации о приложении или статистических данных его работы. В частности: не удалось точно оценить тестируемость в связи с отсутствием тестов, а также сложно корректно оценить отказоустойчивость, так как при работе с программой никогда не возникало проблем с нехваткой оперативной памяти.

1. *Поясните п. 3.4, каким образом это сказывается на показателе комфорта?*
2. *Поясните 2.1 (временные характеристик), о какой нагрузке идет речь и почему это сказывается на данном приложении?*
3. *Поясните 5.3, почему у вас идет речь о сервере?*
4. *У вас в 7.3 написано, что покрытие тестами не выполнено, что вы под этим подразумевали?*
5. *Что собой представляет анализируемость? Что влияет на данную характеристику и упрощает процесс анализа ПО? Приведите примеры.*
6. *Что собой представляет модифицируемость? Что влияет на данную характеристику и упрощает процесс модификации ПО? Приведите примеры.*
7. **Использованные источники**1. Черников Б.В. Оценка качества программного обеспечения: Практикум: учебное пособие / Б.В. Черников, Б.Е. Поклонов / Под ред. Б.В. Черникова. – М.: ИД "ФОРУМ": ИНФРА-М, 2012. – 400 с.: ил.

2. ГОСТ Р ИСО/МЭК 25-10-2015. Системная и программная инженерия. Требования и оценка качества систем и программного обеспечения (SQuaRE).Модели качества систем и программных продуктов. – М.: Стандартинформ, 2015. – 29 с.

**Приложение**

MAIN.CPP

#include "HashMap.h"

#include <iostream>

#include <string>

#include <time.h>

using namespace std;

const int N = 2500;

const int K = 3;

void menu(HashMap<string> &HMap) {

cout << endl

<< "1 - Print\n"

<< "2 - Add element\n"

<< "3 - Delete elem\n"

<< "4 - Export\n"

<< "5 - Exit\n"

<< "Select the menu item: ";

int choice;

cin >> choice;

cin.clear();

cin.ignore(cin.rdbuf()->in\_avail());

switch (choice) {

case 1: {

HMap.print();

menu(HMap);

break;

}

case 2: {

string key;

string val;

cout << "Key: ";

getline(cin, key);

if (key.length() == 6 && isupper(key[0]) && isupper(key[5]) &&

isdigit(key[1]) && isdigit(key[2]) && isdigit(key[3]) &&

isdigit(key[4])) {

cout << "Value: ";

getline(cin, val);

HMap.put(key, val);

}

menu(HMap);

break;

}

case 3: {

string key;

cout << "Key: ";

getline(cin, key);

if (key.length() == 6 && isupper(key[0]) && isupper(key[5]) &&

isdigit(key[1]) && isdigit(key[2]) && isdigit(key[3]) &&

isdigit(key[4])) {

cout << "Elements with collisions: " << HMap.del(key).size() << endl;

}

menu(HMap);

break;

}

case 4: {

HMap.excel(string("excel.txt"));

cout << "OK" << endl;

menu(HMap);

break;

}

case 5: {

break;

}

default:

cout << "Incorrect choice." << endl;

menu(HMap);

}

}

int main() {

srand(time(0));

HashMap<string> HMap(N);

for (int i = 0; i < K; i++) {

string strKey(6, '0');

string strVal(10 + rand() % 70, '\0');

strKey[0] = 'A' + rand() % 26;

strKey[1] = '0' + rand() % 9;

strKey[2] = '0' + rand() % 9;

strKey[3] = '0' + rand() % 9;

strKey[4] = '0' + rand() % 9;

strKey[5] = 'A' + rand() % 26;

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

strVal[i] = (char)'A' + rand() % 26;

HMap.put(strKey, strVal);

}

menu(HMap);

return 0;

}

HASHNODE.H

#pragma once

#include <string>

template <typename V>

class HashNode {

public:

HashNode(const std::string& key, const V& value) : key(key), value(value) {}

std::string getKey() const { return key; }

V getValue() const { return value; }

void setValue(V val) { HashNode::value = val; }

private:

std::string key;

V value;

};

HASHMAP.H

#pragma once

#include "HashNode.h"

#include <fstream>

#include <iostream>

#include <vector>

using namespace std;

template <typename V> class HashMap {

public:

HashMap(int N) {

MAP\_SIZE = N;

map = new HashNode<V> \*[N];

exprt = new int[N];

for (int i = 0; i < N; i++) {

map[i] = nullptr;

exprt[i] = 0;

}

// Generate prime numbers for h2

// 2, 3, 5, 7, 11, 13, 17, 19...N

primes\_h2.push\_back(2);

for (int i = 3; i < N; i++) {

bool prime = true;

for (int j = 0; j < primes\_h2.size() && primes\_h2[j] \* primes\_h2[j] <= i;

j++) {

if (i % primes\_h2[j] == 0) {

prime = false;

break;

}

}

if (prime)

primes\_h2.push\_back(i);

}

};

~HashMap() {

for (int i = 0; i < N; i++)

delete map[i];

delete map;

delete exprt;

};

unsigned int h1(string &key) {

unsigned int h1 = 0;

const int p = 37;

unsigned int p\_pow = 1;

// h(S) = S[0] + S[1] \* P + S[2] \* P ^ 2 + S[3] \* P ^ 3 + ... + S[N] \* P^N

for (size\_t i = 0; i < key.length(); i++) {

h1 += (key[i] - '0' + 1) \* p\_pow;

p\_pow \*= p;

}

return h1;

}

unsigned int h2(string &key) {

unsigned int h2 = 0;

for (size\_t i = 0; i < key.length(); i++)

h2 += key[i] \* key[i];

// Take a random item

return primes\_h2[h2 % primes\_h2.size()];

}

void put(string key, V value) {

unsigned int h1 = HashMap::h1(key);

unsigned int h2 = HashMap::h2(key);

for (size\_t i = 0; i < MAP\_SIZE; i++) {

exprt[h1 % N] += 1;

if (map[h1 % N] == nullptr) {

map[h1 % N] = new HashNode<V>(key, value);

return;

}

else {

if (map[h1 % N]->getKey() == key)

map[h1 % N]->setValue(value);

h1 = (h1 + h2) % N;

}

}

cout << "OVERFLOW" << endl;

}

HashNode<V> \*seach(string &key) {

unsigned int h1 = HashMap::h1(key);

unsigned int h2 = HashMap::h2(key);

for (size\_t i = 0; i < MAP\_SIZE; i++) {

if (map[h1 % N] != nullptr && map[h1 % N]->getKey() == key)

return map[h1 % N];

else

h1 = (h1 + h2) % N;

}

return nullptr;

}

vector<HashNode<V> \*> del(string &key) {

unsigned int h1 = HashMap::h1(key);

unsigned int h2 = HashMap::h2(key);

vector<HashNode<V> \*> collision;

for (size\_t i = 0; i < MAP\_SIZE; i++) {

if (map[h1 % N]->getKey() == key) {

delete map[h1 % N];

map[h1 % N] = nullptr;

return collision;

}

else {

collision.push\_back(map[h1 % N]);

h1 = (h1 + h2) % N;

}

}

}

void excel(string &name) {

ofstream fout(name);

for (size\_t i = 0; i < MAP\_SIZE; i++)

fout << exprt[i] << endl;

fout.close();

}

void print() {

cout << endl;

for (size\_t i = 0; i < MAP\_SIZE; i++)

if (map[i] != nullptr)

cout << i << ": " << map[i]->getKey() << " - " << map[i]->getValue()

<< endl;

}

private:

HashNode<V> \*\*map;

int \*exprt;

int MAP\_SIZE;

vector<int> primes\_h2;

};