Министерство науки и образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Волгоградский государственный технический университет»

КАЧЕСТВО И НАДЕЖНОСТЬ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Программа для построения связей в генеалогическом древе Внешняя спецификация

СОГЛАСОВАНО: Разработчик:

Руководитель проекта: Студент группы

Доцент кафедры ПОАС ПрИн-266

Сычев О. А. Козарез М.В.

« » 2021 г. « » 2021 г.

Нормоконтролер:

Преподаватель кафедры ПОАС

Матюшечкин Д.С.

« » 2021 г.

Волгоград 2021 г.

Оглавление

[1 Проектирование функции translit, решающей главную задачу 3](#_Toc101547056)

[1.1 Модульные тесты 3](#_Toc101547057)

[1.1.1 Реализация модульных тестов 3](#_Toc101547058)

[1.2 Код 5](#_Toc101547059)

[2. Проектирование функции decodeString, решающей главную задачу 5](#_Toc101547060)

[2.1 Код 5](#_Toc101547061)

[3. Проектирование функции tolower 5](#_Toc101547062)

[3.1 Модульные тесты 5](#_Toc101547063)

[3.1.1 Реализация модульных тестов 5](#_Toc101547064)

[3.2 Код 5](#_Toc101547065)

[4. Проектирование функции toupper(char) 6](#_Toc101547066)

[4.1 Модульные тесты 6](#_Toc101547067)

[4.1.1 Реализация модульных тестов 6](#_Toc101547068)

[4.2 Код 6](#_Toc101547069)

[5. Проектирование функции makeLow 6](#_Toc101547070)

[5.1 Модульные тесты 6](#_Toc101547071)

[5.1.1 Реализация модульных тестов 6](#_Toc101547072)

[5.2 Код 6](#_Toc101547073)

[6. Проектирование функции toupper(string) 6](#_Toc101547074)

[6.1 Модульные тесты 6](#_Toc101547075)

[6.1.1 Реализация модульных тестов 6](#_Toc101547076)

[6.2 Код 7](#_Toc101547077)

[7. Проектирование функции inputFile 7](#_Toc101547078)

[7.1 Код 7](#_Toc101547079)

[8. Проектирование функции setDecMap 8](#_Toc101547080)

[8.1 Код 8](#_Toc101547081)

[9. Проектирование функции outputFile 9](#_Toc101547082)

[9.1 Код 9](#_Toc101547083)

# 1 Проектирование функции translit, решающей главную задачу

* 1. Модульные тесты
     1. Реализация модульных тестов

1. TEST\_CLASS(TestLibrary)
2. {
3. public:
5. TEST\_METHOD(\_translitMin)
6. {
7. setDecMap();
8. string eng = "q";
9. vector<string> exp\_string{ "ку" };
10. vector<string> returnString = translit(eng);
11. if (exp\_string == returnString)
12. {
13. Assert::AreEqual(1, 1);
14. }
15. else
16. {
17. Assert::AreEqual(1, 2);
18. }
19. }
20. TEST\_METHOD(\_translitMedium)
21. {
22. setDecMap();
23. string eng = "qqqqq";
24. vector<string> exp\_string{ "кукукукуку" };
25. vector<string> returnString = translit(eng);
26. if (exp\_string == returnString)
27. {
28. Assert::AreEqual(1, 1);
29. }
30. else
31. {
32. Assert::AreEqual(1, 2);
33. }
34. }
35. TEST\_METHOD(\_translitMax)
36. {
37. setDecMap();
38. string eng = "qqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqq";
39. vector<string> exp\_string{ "кукукукукукукукукукукукукукукукукукукукукукукукукукукукукукукукукукукукукукукукукукукукукукукукукуку" };
40. vector<string> returnString = translit(eng);
41. if (exp\_string == returnString)
42. {
43. Assert::AreEqual(1, 1);
44. }
45. else
46. {
47. Assert::AreEqual(1, 2);
48. }
49. }
50. TEST\_METHOD(\_multipleTranslitTwoVariants)
51. {
52. setDecMap();
53. string eng = "ezhik";
54. vector<string> exp\_string{ "езхик", "ежик" };
55. vector<string> returnString = translit(eng);
56. if (exp\_string == returnString)
57. {
58. Assert::AreEqual(1, 1);
59. }
60. else
61. {
62. Assert::AreEqual(1, 2);
63. }
64. }
65. TEST\_METHOD(\_multipleTranslitFourVariants)
66. {
67. setDecMap();
68. string eng = "yozhik";
69. vector<string> exp\_string{ "йозхик", "йожик", "ёзхик", "ёжик" };
70. vector<string> returnString = translit(eng);
71. if (exp\_string == returnString)
72. {
73. Assert::AreEqual(1, 1);
74. }
75. else
76. {
77. Assert::AreEqual(1, 2);
78. }

}

1.2 Код

vector<string> translit(string eng)

{

string copy = makeLow(eng);

decodes.resize(copy.size() + 1);

decodes[copy.size()] = { "" };

vector<string>ru = decodeString(copy, 0);

return ru;

}

2. Проектирование функции decodeString, решающей главную задачу

2.1 Код

vector<string> decodeString(string& line, int pos)

{

if (decodes[pos].size() > 0)

return decodes[pos];

else

{

vector<string>res;

for (int len = 1; (len <= maxCombLen) && (pos + len <= line.size()); ++len)

{

vector<string>& curVars = decMap[line.substr(pos, len)],

leftVars = decodeString(line, pos + len);

if (upper[pos])

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

curVars[i] = toupper(curVars[i]);

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

for (int j = 0; j < leftVars.size(); ++j)

res.push\_back(curVars[i] + leftVars[j]);

}

decodes[pos] = res;

return res;

}

}

3. Проектирование функции tolower

3.1 Модульные тесты

3.1.1 Реализация модульных тестов

TEST\_METHOD(\_tolowerChar)

{

setDecMap();

char symbol = 'K';

char exp\_symbol = 'k';

char returnChar = tolower(symbol);

Assert::AreEqual(exp\_symbol, returnChar);

}

3.2 Код

char tolower(char ch)

{

return static\_cast<char>(tolower(static\_cast<unsigned char>(ch)));

}

4. Проектирование функции toupper(char)

4.1 Модульные тесты

4.1.1 Реализация модульных тестов

TEST\_METHOD(\_toupperChar)

{

setDecMap();

char symbol = 'k';

char exp\_symbol = 'K';

char returnChar = toupper(symbol);

Assert::AreEqual(exp\_symbol, returnChar);

}

4.2 Код

char toupper(char ch)

{

return static\_cast<char>(toupper(static\_cast<unsigned char>(ch)));

}

5. Проектирование функции makeLow

5.1 Модульные тесты

5.1.1 Реализация модульных тестов

TEST\_METHOD(\_makeLow)

{

setDecMap();

string line = "YOZHIK";

string exp\_string = "yozhik";

string returnString = makeLow(line);

Assert::AreEqual(exp\_string, returnString);

}

5.2 Код

string makeLow(string line)

{

upper.resize(line.size());

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

{

char c = tolower(line[i]);

upper[i] = (c != line[i]);

line[i] = c;

}

return line;

}

6. Проектирование функции toupper(string)

6.1 Модульные тесты

6.1.1 Реализация модульных тестов

TEST\_METHOD(\_toupperString)

{

setDecMap();

string line = "yozhik";

string exp\_string = "YOZHIK";

string returnString = toupper(line);

Assert::AreEqual(exp\_string, returnString);

}

6.2 Код

string toupper(string line)

{

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

line[i] = toupper(line[i]);

return line;

}

7. Проектирование функции inputFile

7.1 Код

tring inputFile(string fileName)

{

ifstream fin;

fin.open(fileName);

if (!fin.is\_open())

{

printf("Ошибка при открытии файла!\n");

exit(EXIT\_FAILURE);

}

string line;

int count = 0;

while (!fin.eof())

{

getline(fin, line);

count++;

}

if (count > 1)

{

printf("Количество строк в файле не равно 1!\n");

exit(EXIT\_FAILURE);

}

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

{

if (!((line[i] >= 'A' && line[i] <= 'Z') || (line[i] >= 'a' && line[i] <= 'z') || line[i] == ' '))

{

printf("В исходной строке есть недопустимые символы!\n");

exit(EXIT\_FAILURE);

}

}

cout << line << endl;

fin.close();

return line;

}

8. Проектирование функции setDecMap

8.1 Код

map<string, vector<string>>decMap;

void setDecMap() {

decMap["a"] = { "а" };

decMap["b"] = { "б" };

decMap["c"] = { "ц" };

decMap["d"] = { "д" };

decMap["e"] = { "е" };

decMap["f"] = { "ф" };

decMap["g"] = { "г" };

decMap["h"] = { "х" };

decMap["i"] = { "и" };

decMap["j"] = { "дж" };

decMap["k"] = { "к" };

decMap["l"] = { "л" };

decMap["m"] = { "м" };

decMap["n"] = { "н" };

decMap["o"] = { "о" };

decMap["p"] = { "п" };

decMap["q"] = { "ку" };

decMap["r"] = { "р" };

decMap["s"] = { "с" };

decMap["t"] = { "т" };

decMap["u"] = { "у" };

decMap["v"] = { "в" };

decMap["w"] = { "в" };

decMap["x"] = { "кс" };

decMap["y"] = { "й" };

decMap["z"] = { "з" };

decMap["yo"] = { "ё" };

decMap["yu"] = { "ю" };

decMap["ya"] = { "я" };

decMap["ye"] = { "е" };

decMap["zh"] = { "ж" };

decMap["sh"] = { "ж" };

decMap["kh"] = { "х" };

decMap["eh"] = { "э" };

decMap["ih"] = { "ы" };

decMap["ts"] = { "ц" };

decMap["ch"] = { "ч" };

decMap["A"] = { "А" };

decMap["B"] = { "Б" };

decMap["C"] = { "Ц" };

decMap["D"] = { "Д" };

decMap["E"] = { "Е" };

decMap["F"] = { "Ф" };

decMap["G"] = { "Г" };

decMap["H"] = { "Х" };

decMap["I"] = { "И" };

decMap["J"] = { "Дж" };

decMap["K"] = { "К" };

decMap["L"] = { "Л" };

decMap["M"] = { "М" };

decMap["N"] = { "Н" };

decMap["O"] = { "О" };

decMap["P"] = { "П" };

decMap["Q"] = { "Ку" };

decMap["R"] = { "Р" };

decMap["S"] = { "С" };

decMap["T"] = { "Т" };

decMap["U"] = { "У" };

decMap["V"] = { "В" };

decMap["W"] = { "В" };

decMap["X"] = { "Кс" };

decMap["Y"] = { "Й" };

decMap["Z"] = { "З" };

decMap["Yo"] = { "Ё" };

decMap["Yu"] = { "Ю" };

decMap["Ya"] = { "Я" };

decMap["Ye"] = { "Е" };

decMap["Zh"] = { "Ж" };

decMap["Sh"] = { "Ш" };

decMap["Kh"] = { "Х" };

decMap["Eh"] = { "Э" };

decMap["Ih"] = { "Ы" };

decMap["Ts"] = { "Ц" };

decMap["Ch"] = { "Ч" };

decMap[" "] = { " " };

}

9. Проектирование функции outputFile

9.1 Код

void outputFile(vector<string> ru, string outputFileName)

{

ofstream fout;

fout.open(outputFileName);

if (!fout.is\_open())

{

printf("Ошибка при открытии файла!\n");

exit(EXIT\_FAILURE);

}

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

fout << ru[i] << endl;

fout.close();

}