Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Волгоградский государственный технический университет»

УТВЕРЖДАЮ:

зав. кафедрой ПОАС

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Дворянкин А.М.

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2016 г.

Надежность и качество программного обеспечения

«Определение наличия в графе циклов»

Программа и методика испытаний

СОГЛАСОВАНО: РАЗРАБОТЧИК:

Руководитель работы: Студент группы ПрИн-266

доцент кафедры ПОАС

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Сычев О.А. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Шалиевская А.А.

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2016 г. «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2016 г.

НОРМОКОНТРОЛЛЕР:

доцент кафедры ПОАС

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Сычев О.А.

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2016 г.

2016

Содержание

[1 Объект испытаний 3](#_Toc433899224)

[2 Цель испытаний 3](#_Toc433899225)

[3 Требования к программе 3](#_Toc433899226)

[4 Требования к программной документации 3](#_Toc433899227)

[5 Средства и порядок испытаний 4](#_Toc433899228)

[6 Методы испытаний 4](#_Toc433899229)

[Приложение 1. Тесты для функции search\_cycles. 7](#_Toc433899230)

[Приложение 2. Тесты для функции cycle\_recording 10](#_Toc433899227)

[Приложение 3. Тесты для функции inspection 12](#_Toc433899228)

[Приложение 4. Тесты для функции remove\_cycles 15](#_Toc433899229)

[Приложение 5. Тесты для функции is\_cyclical. 18](#_Toc433899230)

[Приложение 6. Тесты для функции find\_group. 20](#_Toc433899230)

1 Объект испытаний

Программа нахождения и удаления из направленного графа циклов “graph”. Программа предназначена для подготовки графа к укладке на плоскость методом поуровневой укладки.

2 Цель испытаний

Целью испытаний является проверка соответствия программы требованиям к функциональным характеристикам.

3 Требования к программе

Требования к функциональным характеристикам изложены в Техническом задании в пункте 3.1.

4 Требования к программной документации

В бумажной форме должна быть представлена следующая документация:

– техническое задание;

– технический проект в виде описания программы;

– программа и методика испытаний;

– руководство программиста.

В электронной форме должны быть представлены:

– копии всех документов бумажной формы;

– рабочая документация;

– текст программы.

5 Средства и порядок испытаний

Для запуска тестов необходимо подключить библиотеку cfix к проекту программы.

6 Методы испытаний

Перечень проверок функции search\_cycles:

* граф состоит из одной вершины;
* граф состоит из двух вершин;
* граф состоит из двух несвязанных между собой частей;
* в графе нет циклов;
* в графе один цикл;
* в графе несколько циклов;
* в графе все вершины связаны между собой.

Входные данные и соответствующие им ожидаемые результаты для функции search\_cycles представлены в Приложении 1.

Перечень проверок функции cycle\_recording:

* последняя цифра повторяется в цепочке 1 раз;
* последняя цифра повторяется в цепочке 2 раза;
* последняя цифра повторяется в цепочке 3 раза, но один из циклов уже был найден;
* последняя цифра такая же, как и первая.

Входные данные и соответствующие им ожидаемые результаты для функции cycle\_recording представлены в Приложении 2.

Перечень проверок функции inspection:

* цикл уже был найден точно в таком же виде;
* цикл уже был найден самым последним;
* цикл уже был найден, но записан в другом порядке;
* цикл найден впервые, различается только число вершин;
* цикл найден впервые;
* первая вершина проверяемого цикла повторяется в нем, искомый цикл уже был найден;
* первая вершина проверяемого цикла повторяется в нем, искомый цикл найден впервые;
* проверяемый цикл начинается с середины цепочки;
* проверяемый цикл состоит всего из двух вершин;
* ни одного цикла не было найдено ранее;
* длинная часть цепи перед переданной вершиной.

Входные данные и соответствующие им ожидаемые результаты для функции inspection представлены в Приложении 3.

Перечень проверок функции remove\_cycles

* большее количество дуг идет от вершин с большим номером к меньшему;
* большее количество дуг идет от вершин с меньшим номером к большему;
* одинаковое число дуг от вершин с меньшим номером к большему и наоборот;
* для превращения графа в ацикличный достаточно перенаправить 1 дугу;
* для превращения графа в ацикличный достаточно перенаправить 2 дуги;
* для превращения графа в ацикличный достаточно перенаправить 3 дуги;
* для перенаправлений выбрана только одна дуга;
* в графе несколько циклов;
* в графе все вершины связаны между собой.

Входные данные и соответствующие им ожидаемые результаты для функции remove\_cycles представлены в Приложении 4.

Перечень проверок функции is\_cyclical

* есть один цикл;
* есть несколько циклов;
* циклов нет;
* всего две вершины, цикл есть;
* всего две вершины, цикла нет;
* всего одна вершина.

Входные данные и соответствующие им ожидаемые результаты для функции is\_cyclical представлены в Приложении 5.

Перечень проверок функции find\_group

* создать группы по 1 дуге;
* создать группы по 2 дуги;
* создать группы по 3 дуги;
* создать группу из всех элементов;
* в графе всего одна дуга.

Входные данные и соответствующие им ожидаемые результаты для функции find\_group представлены в Приложении 6.

Приложение 1

Тесты для функции search\_cycles

Функция поиска в графе циклов

bool search\_cycles (const QList<QList<int>>&matrix, QList<QList<int>>& cycles);

Входные данные:

matrix –матрица смежности графа

Выходные данные:

cycles – найденные циклы.

Возвращаемое значение:

Найден ли хотя бы один цикл.

|  |  |  |
| --- | --- | --- |
| Входные данные | Ожидаемый результат | |
| 1. Граф состоит из одной вершины(Test\_one\_vertex) | | |
| Screenshot_6 | cycles: - | |
| 1. Граф состоит из двух вершин (Test\_two\_vertices) | | |
| Screenshot_4 | cycles:  {1,2,1} | |
| 1. Граф состоит из двух несвязанных между собой частей(Test\_two\_parts) | | |
| Screenshot_1 | cycles:  {2,4,3,2},  {5,6,5} | |
| Входные данные | | Ожидаемый результат |
| 1. В графе нет циклов(Test\_no\_cycles) | | |
| Screenshot_2 | cycles: - | |
| 1. В графе один цикл(Test\_one\_cycle) | | |
| Screenshot_3 | cycles:  {1,3,4,2,1} | |
| 1. В графе несколько циклов(Test\_multiple\_cycles) | | |
| Screenshot_4 | cycles:  {1,2,1},  {1,2,1,3,1},  {1,3,1},  {1,2,1,3,1,4,1},  {1,3,1,4,1},  {1,4,1},  {1,4,1,2,1} | |
| 1. В графе все вершины связаны между собой(Test\_all\_connected) | | |
| Screenshot_1 | cycles:  {1,2,1},  {1,2,1,3,1},  {1,3,1},  {2,1,3,2},  {3,2,3},  {1,2,1,3,2,3,1}, | |
| Входные данные | Ожидаемый результат | |
|  | {1,3,2,3,1},  {1,2,3,1},  {1,2,3,1,3,2,1}  {1,2,3,2,1},  {1,2,3,2,1,3,1}, | |

Приложение 2

Тесты для функции cycles\_recording

Функция, отбирающая циклы из цепочки посещенных вершин

void searching::cycles\_recording ();

Входные данные:

circuit –цепочка посещенных на текущей ветке вершин

cycles - список ранее найденных вершин

Выходные данные:

cycles – найденные циклы.

|  |  |
| --- | --- |
| Входные данные | Ожидаемый результат |
| 1. Последняя цифра повторяется в цепочке 1 раз(Test\_one\_repeat) | |
| circuit: 2-1-3-4-1  cycles:  {2,1,3,2} | cycles:  {2,1,3,2},  {1,3,4,1} |
| 1. Последняя цифра повторяется в цепочке 2 раза(Test\_two\_repeats) | |
| circuit: 2-1-2-4-1-3-2  cycles:  {3,4,1,3} | cycles:  {3,4,1,3},  {2,1,2,4,1,3,2},  {2,4,1,3,2} |
| 1. Последняя цифра повторяется в цепочке 3 раза, но один из циклов уже был найден(Test\_three\_repeat\_one\_was\_found) | |
| circuit: 2-1-4-1-5-1-3-1  cycles:  {3,4,1,3},  {1,3,1} | cycles:  {3,4,1,3},  {1,3,1},  {1,4,1,5,1,3,1}, |
| Входные данные | Ожидаемый результат |
|  | {1,5,1,3,1} |
| 1. Последняя цифра такая же, как и первая(Test\_last\_digit\_like\_the\_first) | |
| circuit: 2-1-3-4-2  cycles:  {1,2,3,4,1} | cycles:  {1,2,3,4,1},  {2,1,3,4,2} |

Приложение 3

Тесты для функции inspection

Функция, определяющая, был ли найден ранее текущий цикл

bool searching::inspection(int number);

Входные данные:

number-номер вершины в цепочке, с которой начинается проверяемый цикл

circuit –цепочка посещенных на текущей ветке вершин

cycles - список ранее найденных вершин

Возвращаемое значение:

Был ли такой цикл уже найден ранее.

|  |  |  |  |
| --- | --- | --- | --- |
| Входные данные | | | Ожидаемый результат |
| 1. Цикл уже был найден точно в таком же виде(Test\_found\_the\_same) | | | |
| circuit: 1-2-**3-**4-1-3  cycles:  {3,4,1,3},  {1,2,3,4,1},  {1,4,1} | | | true |
| 1. Цикл уже был найден самым последним(Test\_found\_the\_last) | | | |
| circuit: **1-**2-3-1  cycles:  {2,3,4,2},  {1,3,2,1},  {1,2,1},  {1,2,3,1} | | | true |
| Входные данные | | | Ожидаемый результат |
| 1. Цикл уже был найден, но записан в другом порядке(Test\_found\_other\_form) | | | |
| circuit: **4-**3-1-2-4  cycles: {1,2,4,3,1} | | | true |
| 1. Цикл найден впервые, различается только число вершин(Test\_different\_number) | | | |
| circuit: **2**-3-4-2  cycles:  {2,3,2} | | false | |
| 1. Цикл найден впервые(Test\_find\_first\_time) | | | |
| circuit: **1**-2-5-3-1-2-1  cycles:  {1,2,1}  {3,1,2,3}  {5,2,5} | | false | |
| 1. Первая вершина проверяемого цикла повторяется в нем, искомый цикл уже был найден(Test\_vertex\_repeats\_was\_found) | | | |
| circuit: **2**-4-5-3-2-1-2  cycles:  {2,1,2}  {5,3,2,5}  {5,3,2,1,2,4,5} | | true | |
| 1. Первая вершина проверяемого цикла повторяется в нем, искомый цикл найден впервые(Test\_vertex\_repeats\_was\_not\_found) | | | |
| circuit: **2**-4-5-3-2-1-2-4-2  cycles: | | false | |
| Входные данные | | Ожидаемый результат | |
| {2,4,2}  {2,1,2}  {4,5,3,2,1,2,4} | |  | |
| 1. Проверяемый цикл начинается с середины цепочки(Test\_check\_from\_middle) | | | |
| circuit: 2-5-3-**2**-1-3-2  cycles:  {2,1,3,2}  {4,5,3,2,1,2,4} | | true | |
| 1. Проверяемый цикл состоит всего из двух вершин(Test\_checked\_consists\_of\_two\_vertex) | | | |
| circuit: **1**-2-1  cycles:  {1,2,1} | | true | |
| 1. Ни одного цикла не было найдено ранее(Test\_no\_found\_cycles) | | | |
| circuit: **1**-2-4-1  cycles: | | false | |
| 1. Длинная часть цепи перед переданной вершиной(Test\_long\_part\_before) | | | |
| circuit: 5-3-1-5-2-**1**-2-4-1  cycles:  {1,2,4,1} | true | | |

Приложение 4

Тесты для функции remove\_cycles

void remove\_cycles (QList<QList<int>>& matrix, QList<QPair <int,int>> & arc);

Входные данные:

matrix – матрица смежности графа

Выходные данные:

arc- список перенаправленных дуг

|  |  |  |
| --- | --- | --- |
| Входные данные | | Ожидаемый результат |
| 1. Большее количество дуг идет от вершин с большим номером к меньшему(Test\_from\_high\_to\_low) | | |
| Screenshot_2 | | {1,3} или {2,4} или {4,1} или {3,2} |
| 1. Большее количество дуг идет от вершин с меньшим номером к большему(Test\_from\_low\_to\_high) | | |
| Screenshot_1 | | {5,1} или {2,5} |
| 1. Одинаковое число дуг от вершин с меньшим номером к большему и наоборот(Test\_no\_prevailing\_direction) | | |
| Входные данные | | Ожидаемый результат |
| Screenshot_1 | | {3,4} |
| 1. Для превращения графа в ацикличный достаточно перенаправить 1 дугу(Test\_redirect\_one) | | |
| Screenshot_1 | {4,2} | |
| 1. Для превращения графа в ацикличный достаточно перенаправить 2 дуги(Test\_redirect\_two) | | |
| Screenshot_4 | {4,1},{5,3}  или {4,1},{3,4}  или {4,1},{4,5}  или {2,4},{3,4}  или {1,2},{3,4} | |
| 1. Для превращения графа в ацикличный достаточно перенаправить 3 дуги(Test\_redirect\_three) | | |
| Screenshot_5 | {1,3},{2,4},{5,6}  или {1,3},{2,4},{6,5}  или {3,1},{2,4},{5,6}  или {3,1},{2,4},{6,5}  или {4,3},{1,3},{5,6}  или {1,3},{4,3},{6,5}  или {3,2},{3,1},{5,6}  или {3,2},{3,1},{6,5}  или {3,2},{1,3},{6,5} | |
| Входные данные | Ожидаемый результат | |
|  | или {3,2},{1,3},{6,5} | |
| 1. В графе несколько циклов(Test\_few\_cycles) | | |
| Screenshot_4 | {1,2},{1,3},{1,4}  или {1,2},{1,3},{4,1}  или {1,2},{3,1},{1,4}  или {1,2},{3,1},{4,1}  или {2,1},{1,3},{1,4}  или {2,1},{1,3},{4,1}  или {2,1},{3,1},{1,4}  или {2,1},{3,1},{4,1} | |
| 1. В графе все вершины связаны между собой(Test\_all\_connected) | | |
| Screenshot_1 | {1,2},{1,3},{2,3}  или {1,2},{1,3},{3,2}  или {1,2},{3,1},{2,3}  или {2,1},{1,3},{3,2}  или {2,1},{3,1},{2,3}  или {2,1},{3,1},{3,2} | |

Приложение 5

Тесты для функции is\_cyclical

Функция, проверяющая, является ли граф цикличным

bool is\_cyclical (QList<QList<int>>& matrix);

Входные данные:

matrix – матрица смежности графа с перенаправленными дугами

Возвращаемое значение:

Есть ли в графе циклы

|  |  |  |  |
| --- | --- | --- | --- |
| Входные данные | | | Ожидаемый результат |
| 1. Есть один цикл(Test\_one\_cycle) | | | |
| Screenshot_1 | | | true |
| 1. Есть несколько циклов(Test\_multiple\_cycles) | | | |
| Screenshot_2 | | | true |
| 1. Циклов нет(Test\_no\_cycles) | | | |
| Screenshot_3 | | | false |
| 1. Всего две вершины, цикл есть(Test\_two\_vertices\_cycle) | | | |
| Входные данные | Ожидаемый результат | | |
| Screenshot_4 | | true | |
| 1. Всего две вершины, цикла нет(Test\_two\_vertices\_no\_cycle) | | | |
| Screenshot_5 | | false | |
| 1. Всего одна вершина(Test\_one\_vertex) | | | |
| Screenshot_6 | | false | |

Приложение 6

Тесты для функции find\_group

Функция, находящая все возможные группы дуг по i элементов в каждой

void find\_group (const QList<QList<int>>&arcs, int i, QList< QSet<QPair<int,int>>> &groups);

Входные данные:

arcs - дуги, из которых необходимо сформировать группы по i элементов

i-количество элементов, которое должно быть в каждой сформированной группе

groups-контейнер, в котором хранятся группы по i-1 элементов, или пустой контейнер, если требуется создать группы по 1 элементу.

Выходные данные:

groups- контейнер, в каждом элементе которого хранится группа из i дуг в каждой

|  |  |  |
| --- | --- | --- |
| Входные данные | | Ожидаемый результат |
| 1. Создать группы по 1 дуге(Test\_one\_arc\_in\_the\_group) | | |
| arc: {{1,2},{2,3},{2,4},{1,4}}  i=1;  groups: - | | groups:  {1,2},  {2,3},  {2,4},  {1,4}. |
| 1. Создать группы по 2 дуги(Test\_two\_arcs\_in\_the\_group) | | |
| arc: {{1,2},{2,3},{2,4},{1,4}}  i=2;  groups:  Входные данные | | groups:  {{1,2},{2,3}},  {{1,2},{2,4}},  Ожидаемый результат |
|  | | {{1,2},{1,4}},  {{2,3},{2,4}},  {{2,3},{1,4}},  {{2,4},{1,4}} |
| 1. Создать группы по 3 дуги(Test\_three\_arcs\_in\_the\_group) | | |
| arc: {{1,2},{2,3},{2,4},{1,4}}  i=3;  groups: | | groups:  {{1,2},{2,3},{2,4}},  {{1,2},{2,3},{1,4}},  {{1,2},{2,4},{1,4}},  {{2,3},{2,4},{1,4}} |
| 1. Создать группу из всех элементов(Test\_all\_arcs\_in\_the\_group) | | |
| arc: {{1,2},{2,3},{2,4},{1,4}}  i=4;  groups: | groups:  {{1,2},{2,3},{2,4},{1,4}} | |
| 1. В графе всего одна дуга(Test\_only\_one\_arc) | | |
| arc:{1,2}  i=1;  groups: | groups:  {1,2} | |