**БГУИР**

**Кафедра ИИТ**

**Отчет по лабораторной работе №2**

**Тема: «Алгоритмы операций над графиками и их реализация»**

Выполнил:

Студент группы 421702 Мяло В. В.

Проверил:

Рудьман И.О.

Минск

2024

**Цель работы:**

Научиться составлять алгоритм выполнения операций над графиками.

**Задача:**

Выполнить все основные операции над графиками.

**Уточнение постановки задачи:**

Ограничения на элементы множеств и графиков: элементы должны быть целыми числами в диапазоне [-2³¹; 2³¹-1]. Ограничений на область определения и область значений нет.

**Определения и понятия, использовавшиеся в ходе выполнения работы:**

1.График — это множество пар, т.е. множество, каждый элемент которого

является парой или кортежем длины 2. Множество Р называется графиком, если

каждый его элемент — пара.

2. Областью определения графика Р называется множество пр1P (проекция

на первую ось (ось абсцисс) данного графика).

3. Областью значения графика называется множество проекций на вторую

ось (ось ординат) (пр2Р).

4. Инверсией графика Р называют множество инверсий пар из Р.

5. График Р называется симметричным, если он наряду с любой своей

парой содержит ее инверсию.

6. Пусть М — произвольное множество. Тогда считают ΔM — множество

всех пар вида <х, х>, где х присутствует во всем множестве М. ΔM является симметричным графиком и называется *диагональю*.

7. Композиция графиков Р и Q – график Р◦Q, в котором упорядоченные пары удовлетворяют следующему свойству: существует элемент z такой, что выполняется <x, z> P и <z, y> Q: Р◦Q = {<x, y>|<x, z> P и <z, y> Q}.

**Алгоритмы для упрощения реализации программы:**

1. Проверка наличия некоторой пары <a, b> в графике P.

1.1. Берем пару <c, d> графика P. Далее см. п. 1.2.

1.1.1. Берем следующую пару <c, d> графика P.

1.2. Проверяем пару <c, d> на предмет совпадения с парой <a, b>

1.2.1. Если пара <c, d> совпадает с парой <a, b> (c=a и d=b), то см. п. 1.3.

1.2.2. Если пара <c, d> не совпадает с парой <a, b> (c≠a и d≠b) и не является последней в графике, см. п. 1.1.1.

1.2.2.1. Если пара <c, d> не совпадает с парой <a, b> (c≠a и d≠b) и является последней в графике P, то пара <a, b> не найдена в графике P.

1.3. Пара <a, b> найдена в графике P.

2. Проверка наличия некоторого элемента E в множестве X.

6.1. Берем элемент множества X. Далее смотреть пункт 6.2.

6.1.1. Берем следующий элемент множества X.

6.2. Проверяем данный элемент на совпадение с E.

6.2.1. Если данный элемент совпадает с E, то переходим к пункту 6.3.

6.2.2. Если он не совпадает с E и не является последним, то см. п. 6.1.1.

6.2.3. Если элемент является последним, то элемент не найден.

6.3. Элемент найден

**Алгоритм реализации операций:**

1. Операция инверсии графика P. Алгоритм реализации данной операции следующий:

1.1. Создаем пустой график Q, в который в будущем будет помещен результат выполнения операции инверсии графика P.

1.2. Берем пару <a, b> графика P. Далее см. п. 1.3.

1.2.1. Берем следующую пару <a, b> графика P.

1.3. Добавляем пару <b, a> к графику Q.

1.3.1. Если пара последняя в графике P, см. п. 1.4.

1.3.2. Иначе, см. п. 1.2.1.

1.4. Инверсия графика P составлена.

2. Операция проверки графика P на симметричность. Алгоритм таков:

2.1. Берем пару <a, b> графика P. Далее см. п. 2.2.

2.1.1. Берем следующую пару <a, b> графика P.

2.2. Проверяем наличие пары <b, a> в графике P (подробнее — см п.1 раздела «Алгоритмы для упрощения реализации программы).

2.2.1. Если пара <b, a> не была найдена в графике P, то см. п. 2.3.

2.2.2. Если пара <b, a> была найдена в графике P и при этом пара <a, b> не является последней в графике P, то см. п. 2.1.1.

2.2.2.1. Если пара <b, a> была найдена в графике P и при этом пара <a, b> является последней в графике P, то график P является симметричным.

2.3. График P не является симметричным.

3. Операция нахождения диагонали от множества M:

3.1. Создаем пустой график A=ΔM, в котором будет находиться диагональ.

3.2. Берем элемент E множества M. Далее см. п. 3.3.

3.2.1. Берем следующий элемент E множества M.

3.3. Добавляем пару <E, E> в график A.

3.3.1. Если элемент Е последний в множестве, см. п. 3.4.

3.3.2. Иначе, см. п. 3.2.1.

3.4. Диагональ составлена.

4. Операция нахождения композиции графиков P и Q. Алгоритм следующий:

4.1. Создаем пустой график R, в котором в будущем будет помещен результат выполнения операции композиции.

4.2. Берем пару <a, b> графика P. Далее см. п. 4.2.2.

4.2.1. Берем следующую пару <a, b> графика P.

4.2.2. Берем пару <c, d> графика Q. Далее см. п. 4.2.2.2.

4.2.2.1. Берем следующую пару <c, d> графика Q.

4.2.2.2. Если b≠c, то см. п. 4.2.2.1.

4.2.2.2.1. Иначе проверяем наличие пары <a, d> в графике R (подробнее см. п. 1 раздела «Алгоритмы для упрощения реализации программы»)

4.2.2.3. Если пара <a, d> не была найдена в графике R, то добавляем ее в график.

4.2.2.4. Если пара <c, d> является последней парой в графике Q, то см. п 4.2.3.

4.2.2.4.1. Иначе, см. п. 4.2.2.1.

4.2.3. Если пара <a, b> является последней парой в графике P, то см. п. 4.3.

4.2.3.1. Иначе, см. п. 4.2.1.

4.3. Композиция графиков P и Q найдена.

5. Операция нахождения области определения графика G. Алгоритм:

5.1. Создаем пустое множество D, в котором будет храниться область определения графика G.

5.2. Берем пару <x, y> из графика G.

5.2.1. Берем следующую пару <x, y> из графика G.

5.2.2. Проверяем наличие элемента x в множестве D (см. п. 2 раздела «Алгоритмы для упрощения реализации программы»).

5.2.2.1. Если элемент x не был найден, то добавляем его.

5.2.3. Если пара <x, y> не является последней, то см. п. 5.2.1.

5.3. Область определения графика G найдена.

6. Операция нахождения области значений графика G. Алгоритм:

6.1. Создаем пустое множество E, в котором будет храниться область значений графика G.

6.2. Берем пару <x, y> из графика E.

6.2.1. Берем следующую пару <x, y> из графика E.

6.2.2. Проверяем наличие элемента y в множестве E (см. п. 2 раздела «Алгоритмы для упрощения реализации программы»).

6.2.2.1. Если элемент y не был найден, то добавляем его.

6.2.3. Если пара <x, y> не является последней, то см. п. 5.2.1.

6.3. Область значений графика G найдена.

**Алгоритм выполнения программы:**

1. Сначала пользователю предоставляется право выбрать операцию.

2. Далее, в зависимости от операции, пользователю будет необходимо ввести данные для выполнения выбранной им операции.

3. Затем происходит обработка введенного пользователем требования.

4. Результат выполнения выбранной пользователем операции выводится в консоль.

**Вывод:**

В ходе выполнения данной лабораторной работы я научился составлять алгоритм выполнения некоторых операций над графиками и сделал это, использовав язык программирования C++.