|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ **Информатика и системы управления**

КАФЕДРА **Компьютерные системы и сети (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.01 Информатика и вычислительная техника**

**Отчет**

|  |  |
| --- | --- |
| ***по лабораторной работе №*** | *2* |

**Название:**

*Тестирование и повышение качества программ*

***Дисциплина:*** *Технология разработки программных систем*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Студент* | *ИУ6-42Б* |  |  | *М.Э. Хабаров* |
|  | *(Группа)* |  | *(Подпись, дата)* | *(И.О. Фамилия)* |
|  |  |  |  |  |
| *Преподаватель* |  |  |  | *Е.К.Пугачёв* |
|  |  |  | *(Подпись, дата)* | *(И.О. Фамилия)* |

*Москва, 2021*

***Вариант 15***

***ВВЕДЕНИЕ***

Одним из наиболее трудоемких этапов (от 30 до 60 % общей трудоемкости) создания программного продукта является тестирование. Причем доля стоимости тестирования в общей стоимости разработки имеет тенденцию возрастать при увеличении сложности комплексов программ и повышении требований к их качеству. В связи с этим большое внимание уделяется выбору стратегии и методов тестирования, что не является тривиальной задачей. Таким образом, при подготовке к тестированию необходимо ответить на следующие вопросы:

* Какую стратегию тестирования выбрать и почему, как ее реализовать;
* Какой из методов выбранной стратегии тестирования выбрать и почему;
* Как грамотно подготовить тестовый набор данных и сколько тестов необходимо разработать?

***Цель работы:***

* Приобрести навыки тестирования схем алгоритмов, исходных кодов программ и исполняемых модулей.

***Основная часть***

***Структурный контроль***

***Задача:***  Программа должна определять корни уравнений с помощью “Метода хорд”.

***Листинг программы для тестирования:***

1 #include <iostream>

2 using namespace std;

3 float f( float x, float y)

4 { return x\*x-2; }

5 int main()

6 {

7 float x, p, b, eps;

8 cin>>x; //начальное значение

9 cin>>b; //конечное значение

10 cin>>eps; //точность

11 do

12 { p=x;

13 x=x-f(x)\*(b-x)/(f(b)-f(x));

14 } while (x-p<=eps);

15 cout<<” x = “<<x<<” y = ”<<f(x);

16 }

Ответим на вопросы и заполним таблицу:

1. Обращение к данным

* Все ли переменные инициализированы? ДА, переменная ‘p’ – лишняя.
* Не превышены ли максимальные (или реальные) размеры массивов и строк? НЕТ
* Не перепутаны ли строки со столбцами при работе с матрицами? НЕТ
* Присутствуют ли переменные со сходными именами? НЕТ
* Используются ли файлы? НЕТ
* Использованы ли нетипизированные переменные, открытые массивы, динамическая память? НЕТ

1. Вычисления

* Правильно ли записаны выражения? НЕТ, некорректные выражения в строках 12,14
* Корректно ли производятся вычисления неарифметических переменных? ДА
* Корректно ли выполнены вычисления с переменными различных типов? НЕТ, отсутствует модуль в строке 14
* Возможны ли переполнение разрядной сетки или ситуация машинного нуля? НЕТ
* Соответствуют ли вычисления заданным требованиям точности? ДА
* Присутствуют ли сравнения переменных различных типов? НЕТ

1. Передача управления

* Будут ли корректно завершены циклы? ДА
* Будет ли завершена программа? ДА
* Существуют ли циклы, которые не будут выполняться из-за нарушения условий входа? НЕТ
* Существуют ли поисковые циклы? ДА

1. Интерфейс

* Соответствуют ли списки параметров и аргументов по порядку, типу, единицам измерения? НЕТ, лишний формальный параметр “y” в заголовке функции
* Не изменяет ли подпрограмма аргументов, которые не должны изменяться? НЕТ
* Не происходит ли нарушения области действия глобальных и локальных переменных с одинаковыми именами? НЕТ
* Соответствует ли выводимая информация требованиям задачи? НЕТ

Таблица - Результат структурного контроля

|  |  |  |  |
| --- | --- | --- | --- |
| **№ вопроса** | **Строки, подлежащие проверке** | **Результат проверки** | **Вывод** |
| 1.1 | 7 | float x, p, b, eps; | Все переменные инициализированы, но переменная “p” является лишней из-за некорректности выражения в строке 12 |
| 2.1 | 12, 14 | 12: p=x;  14: while (x-p<=eps); | В 12й строке некорректное выражение, на месте которого должно быть: b = b - f(b) \* (x - b) / (f(x) - f(b));  В 14й строке некорректная операция сравнения, на месте которой должно быть: while (abs(x-b)>eps); |
| 2.3 | 14 | while (x-p<=eps); | В данной строке забыт модуль в операции сравнения: while (abs(x-b)>eps); |
| 4.1 | 3 | float f( float x, float y) | В заголовке функции объявлен лишний формальный параметр “y” |
| 4.4 | 15 | cout<<” x = “<<x<<” y = ”<<f(x); | В данной строке присутствует вывод значения “y”, которое не требуется по условию задачи. |

***Корректный вариант программы:***

#include <iostream>

#include <cmath>

using namespace std;

float f(float x)

{

return x \* x - 2;

}

int main()

{

float x, p, b, eps;

cin >> x; //начальное значение

cin >> b; //конечное значение

cin >> eps; //точность

do

{

b = b - f(b) \* (x - b) / (f(x) - f(b));

x = x - f(x) \* (b - x) / (f(b) - f(x));

} while (abs(x - b) > eps);

cout <<" x = "<< x;

}

***Вывод по структурному контролю:*** Структурный контроль позволят выявить ошибки на ранних этапах разработки, когда программа закодирована, но тестирование на машине ещё не началось. Также следует отметить, что в данном методе тестирование должно осуществляться группой специалистов, в которую входят автор программы, проектировщик, специалист по тестированию и координатор (компетентный программист, но не автор программы).

***Принцип “белого ящика”***

Проведем тестирование блок-схемы заданной по принципу “белого ящика”.

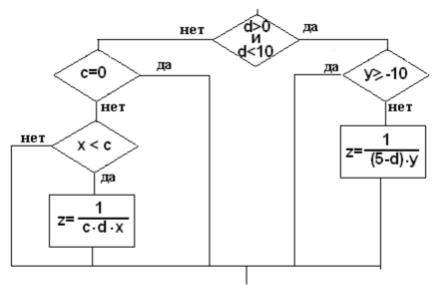


Таблица – Покрытие операторов.

|  |  |  |  |
| --- | --- | --- | --- |
| **Номер теста** | **Назначение теста** | **Значения исходных данных** | **Ожидаемый результат** |
| 1 | Покрытие z=1/(c\*d\*x) | x=0;y=0;c=2;d=10  или  x=1;y=0;c=2;d=0 | Ошибка |
| 2 | Покрытие  z=1/((5-d)\*x) | x=1;y=-11;c=2;d=5 | Ошибка |

***Вывод:*** В схеме два оператора. Их можно покрыть двумя тестами. Тесты 1 и 2 показывают, что возможно прерывание.

Таблица – Покрытие решений.

|  |  |  |  |
| --- | --- | --- | --- |
| **Номер теста** | **Назначение теста** | **Значения исходных данных** | **Ожидаемый результат** |
| 1 | Путь Нет-Нет-Нет | x=5;y=0;c=2;d=10 | Вычислений нет |
| 2 | Путь Нет-Нет-Да | x=0;y=0;c=2;d=0 | Ошибка |
| 3 | Путь Нет-Да | x=5;y=0;c=0;d=10 | Вычислений нет |
| 4 | Путь Да-Да | x=1;y=10;c=2;d=5 | Вычислений нет |
| 5 | Путь Да-Нет | x=1;y=-11;c=2;d=5 | Ошибка |

***Вывод:*** Критерий покрытия решений удовлетворяет критерию покрытия операторов, но является более сильным. Тестирование показывает, что все решения покрываются и “мертвые” ветви отсутствуют. Тесты 2, 5 показывают, что возможно прерывание.

Таблица – Комбинаторное покрытие условий.

|  |  |  |  |
| --- | --- | --- | --- |
| **Номер теста** | **Назначение теста** | **Значения исходных данных** | **Ожидаемый результат** |
| 1 | Комбинация  d∈(-∞;0]∪[10;+∞);c=0 | c=0;d=10 | Вычислений нет |
| 2 | Комбинация  d∈(-∞;0]∪[10;+∞);c!=0;x<c | x=0;y=0;c=2;d=0 | Ошибка |
| 3 | Комбинация  d∈(-∞;0]∪[10;+∞);c!=0;x>=c | x=5;y=0;c=2;d=10 | Вычислений нет |
| 4 | Комбинация  0<d<10;y>=-10 | x=1;y=10;c=2;d=5 | Вычислений нет |
| 5 | Комбинация  0<d<10;y<-10 | x=1;y=-11;c=2;d=5 | Ошибка |

Комбинации {d>0 и d>=10},{d<=0 и d<10},{d<=0 и d>=10} представимы в виде:

d∈(-∞;0]∪[10;+∞)

***Вывод:*** Критерий комбинаторного покрытия условий показал, что в тесте 2 может возникнуть прерывание в случаях х=0 или d=0, а в тесте 5 будет ошибка при d=5.

***Вывод по “белому ящику”:*** Стратегия тестирования по принципу “белого ящика” позволяет проверить внутреннюю структуру программы. Но метод имеет ряд недостатков:

* Не обнаруживает пропущенные маршруты;
* Не обнаруживает ошибок, появление которых зависит от обрабатываемых данных (например, if (a-b)<eps — пропуск функции abs проявится, только если a < b);
* Не дает гарантии, что программа соответствует описанию (например, если вместо сортировки по убыванию написана сортировка по возрастанию).

***Принцип “черного ящика”***

Программа должна осуществлять управление шаром.

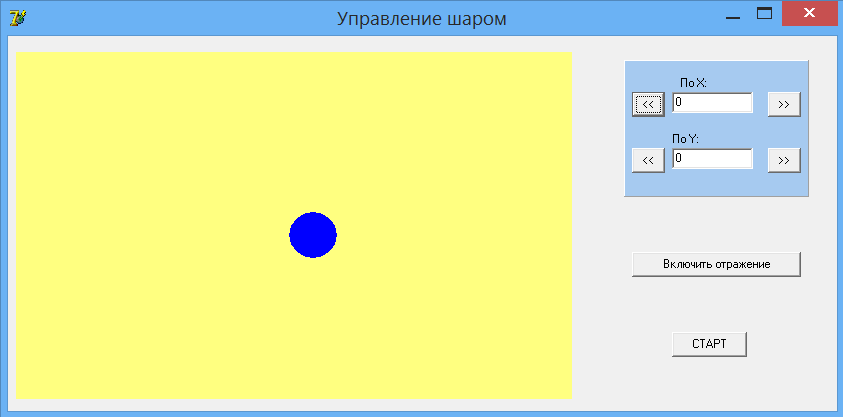


Рисунок 1 – Интерфейс приложения

В данном приложении не реализована возможность редактирования полей X, Y с клавиатуры, поэтому метод классов эквивалентности неприменим.

Анализ граничных значений невозможен, так как исходные данные определены на всей области Z целых чисел.

Таблица –Анализ причинно-следственных связей.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **Значения исходных данных** | **Ожидаемый результат** | **Реакция программы** | **Вывод** |
| 1 | x<0 | Шар двигается влево | Шар двигается влево | Программа отработала верно |
| 2 | x>0 | Шар двигается вправо | Шар двигается вправо | Программа отработала верно |
| 3 | y<0 | Шар двигается вниз | Шар двигается вверх | Программа отработала неверно |
| 4 | y>0 | Шар двигается вверх | Шар двигается вниз | Программа отработала неверно |
| 5 | x=0 y=0 | Шар не двигается | Шар не двигается | Программа отработала верно |

Таблица – Предположения об ошибке.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **Назначение**  **теста** | **Исходные данные** | **Реакция программы** | **Вывод** |
| 1 | Проверка выхода шара за область с включенным отражением | y=0  На границе с областью резко изменять значения х | Шар начинает колебаться практически на месте и выходит из зоны | Программа отработала неверно |
| 2 | Проверка работы кнопки “Включить отражение” | Нажатие на кнопку “Включить отражение” | При первом нажатии на кнопку её текст не изменяется. В дальнейшем возникает ситуация, что при включении отражения мы нажимаем на кнопку “Отключить отражение” | Программа отработала  Верно, но с некорректным функционалом интерфейса |
| 3 | Проверка работы кнопок для изменения значений полей X и Y | Нажатие кнопок для изменения значений полей X и Y | Значения полей изменяются некорректно | Программа отработала, но с некорректной работой ввода |
| 4 | Проверка реакции шара на отключение отражения и включение отражения | Нажатие кнопки отключения отражения | Если отключить отражение, то шар выйдет за пределы области. После этого, если нажатие кнопки включения отражения совпадает с положением шара вне области сверху, снизу, слева, справа от начальной области, то шар залипнет | Программа отработала некорректно |
| 5 | Проверка соблюдения области при движении шара | По умолчанию | Шар частично выходит за границы области слева, справа и сверху | Программа отработала верно, но в процессе движения шар не соблюдал границы области |
| 6 | Проверка возможности ввода с клавиатуры | x=2 y=3 | Программа не воспринимает данные, которые вводятся в поля X,Y вручную | Программа отработала, но без возможности ввода с клавиатуры |

***Вывод по “черному ящику”:*** Это тестирование с управлением по данным. Здесь программа рассматривается как «черный ящик» и тестирование выявляет несоответствие программы спецификации.