

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ» (ДГТУ)**

Факультет Информатика и вычислительная техника

(наименование факультета)

Кафедра Кибербезопасность информационных систем

(наименование кафедры)

**Практическое занятие № 6**

на тему «Исследование моделей надежности программного

обеспечения с использованием моделей Шумана и Миллса»

Выполнил обучающийся гр. ВКБ42

Михайлов А.С.

Проверил:

доцент кафедры «КБИС»

Куликова Ольга Витальевна

доцент кафедры

Скляров А.В.

Ростов-на-Дону

2024

**Цель работы:**

Изучение методик определения надежности программного обеспечения.

**Задание 1:**

Программа содержит *I* командных строк, из них, до начала эксплуатации (после периода отладки), *M* командных строк содержат ошибки. После *t* дней работы обнаружено *M1* ошибок. Коэффициент пропорциональности равен 0,7.

**Вариант 8:**

1. Для заданных исходных данных, определить основные характеристики надежности программы (задание по вариантам приведено в таблице 1, номер варианта устанавливается преподавателем).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № варианта | Кол-во машинных команд (строк),  I | Кол-во ошибок после отладки,  M | Кол-во дней тестирования,  t | Кол-во обнаруженных ошибок после тестирования, M1 |
| 8 | 2930 | 40 | 82 | 18 |

2. Рассчитать следующие показатели надежности:

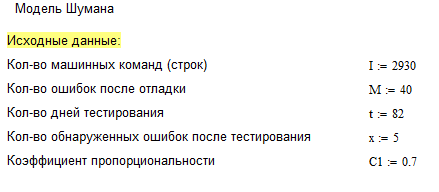
- среднее время безошибочной работы программы,

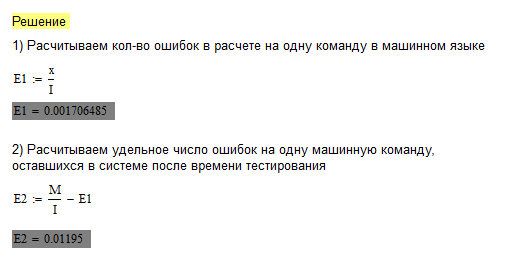
- интенсивность отказов программы,

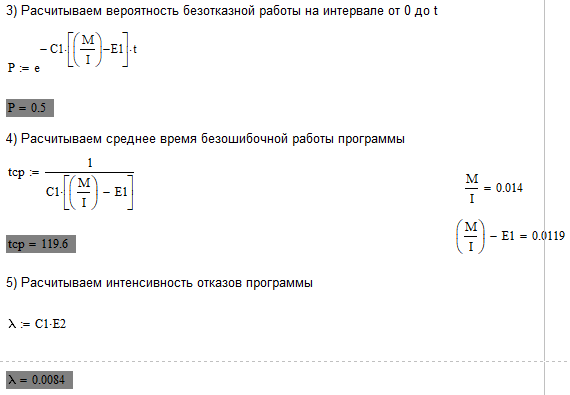
- вероятность безотказной работы программы.

3. Оформить отчет о проделанной работе:

**Ход работы:**







**Вывод:** Таким образом из анализа ПО с помощью модели Шумана можно сделать вывод, что данное программное средство обладает низким количеством ошибок на одну машинную команду, при этом вероятность отказа программного средства выше 50% на интервале от 0 до t.

**Задание 2:**

В программу преднамеренно внесли *W* ошибок. В результате тестирования обнаружено *V* ошибок, из которых *W* ошибок были внесены преднамеренно. Все обнаруженные ошибки исправлены. До начала тестирования предполагалось, что программа содержит не более *R* ошибок.

**Вариант 8:**

1. Для заданных исходных данных, определить основные характеристики надежности программы (задание по вариантам приведено в таблице 2, номер варианта устанавливается преподавателем).

|  |  |  |  |
| --- | --- | --- | --- |
| № варианта | Кол-во преднамеренно внесенных ошибок,  W | Кол-во ошибок, обнаруженных в результате тестирования,  V | Кол-во предполагаемых ошибок,  R |
| 8 | 20 | 21 | 3 |

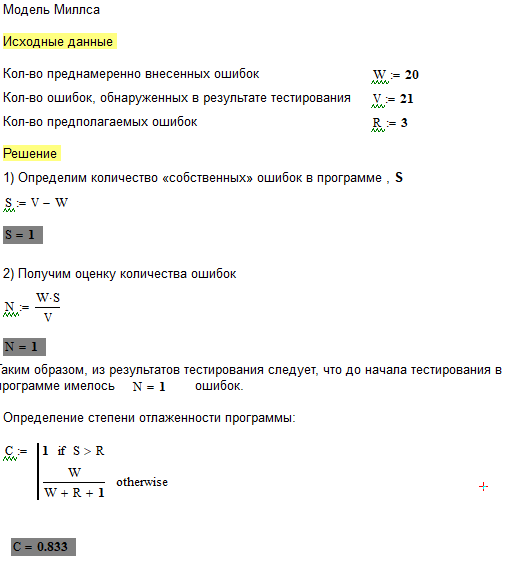
2. Рассчитать следующие показатели надежности:

- количество ошибок до начала тестирования,

- степень отлаженности программы.

3. Оформить отчет о проделанной работе.

**Ход работы:**



**Вывод:** Таким образом из проделанного анализа программного обеспечения можно сделать вывод. что данное ПО обладает высокой степенью отлаженности программы более 70% и низким количеством ошибок.

**Задание 3:**

В программу было преднамеренно внесено W ошибок. В процессе четырех тестов было выявлено определенное количество ошибок.

**Вариант 4:**

1. Для заданных исходных данных, определить основные характеристики надежности программы (задание по вариантам приведено в таблице 3, номер варианта устанавливается преподавателем).

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № варианта | Кол-во преднамеренно внесенных ошибок,  W | Кол-во предполагаемых ошибок,  R | Номер теста | 1 | 2 | 3 | 4 |
| 4 | 15 | 5 | V | 6 | 4 | 3 | 2 |
|  | S | 3 | 1 | 1 | 0 |

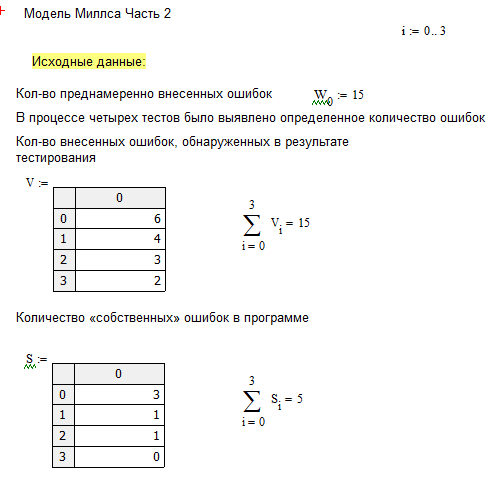
2. Рассчитать следующие показатели надежности:

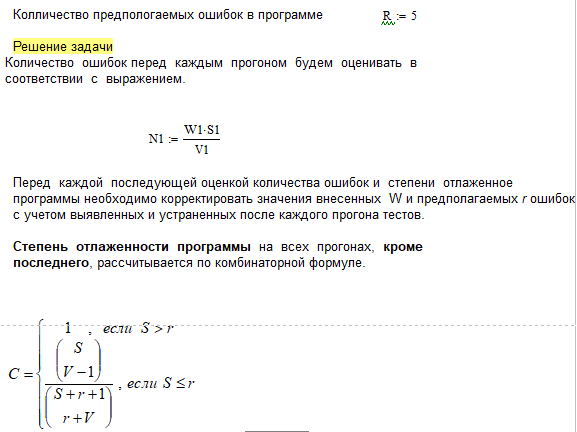
- количество ошибок до начала теста,

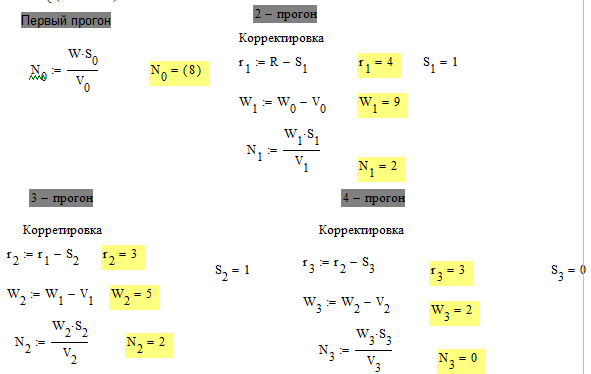
- степень отлаженности программы после каждого теста;

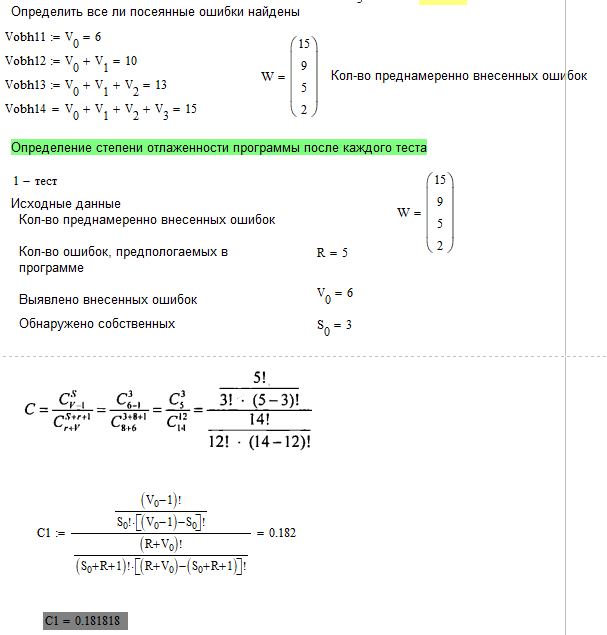
- построить диаграмму зависимости возможного числа ошибок в данной программе от номера теста.

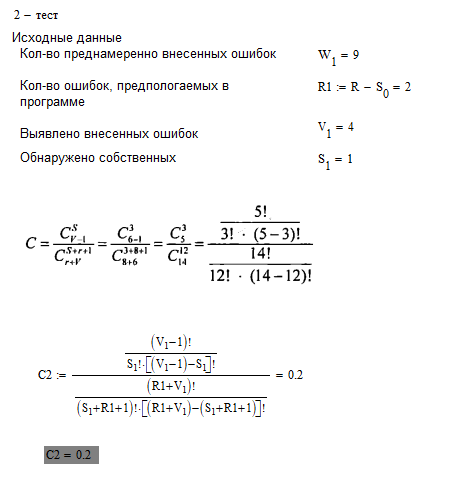
**Ход работы:**

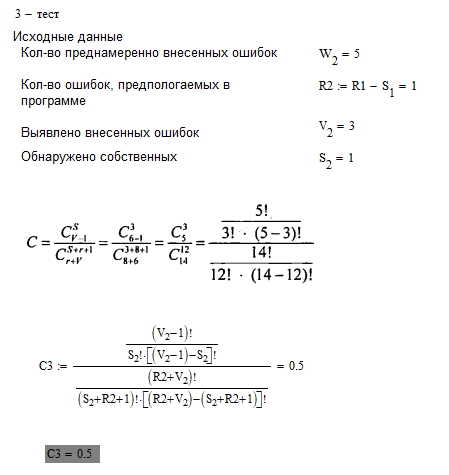


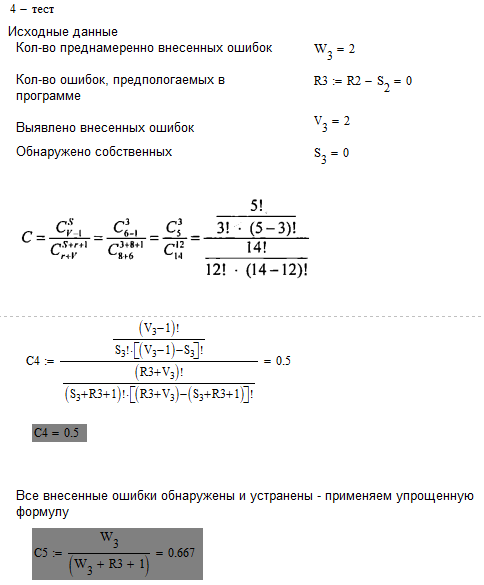










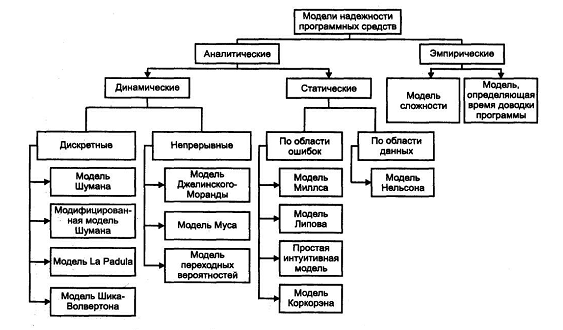


**Вывод:** Таким образом из проделанного анализа можно сделать вывод, что данное надежность программного средства увеличивается после каждого прохода устранения ошибок и повторного тестирования ПО, при этом в конце после устранения всех ошибок программа имеет наивысшую степень надёжности равную 68%.

**Контрольные вопросы:**

1. Как классифицируются основные модели надежности программных средств.

Модель надежности программного обеспечения - это математическая модель, построенная для оценки зависимости надежности программного обеспечения от некоторых определенных параметров.Основные модели надежности программных средств представлены на рис. 1



1. На каких допущениях основана модель Шумана?

Модель Шумана строится на основе нескольких критериев:

- общее число команд в программе на машинном языке постоянно;

- в начале компоновочных испытаний число ошибок равно некоторой постоянной величине, и по мере исправления ошибок их становится меньше. В ходе испытаний программы новые ошибки не вносятся;

- ошибки изначально различимы, по суммарному числу исправленных ошибок можно судить об оставшихся;

- интенсивность отказов программы пропорциональна числу остаточных ошибок.

1. Как рассчитывается вероятность безотказной работы программы в модели Шумана?

Предполагается, что до начала тестирования (т.е. в момент τ=0) имеется M ошибок. В течение времени тестирования τ обнаруживается ε1(τ) ошибок в расчете на одну команду в машинном языке.

Тогда удельное число ошибок на одну машинную команду, оставшихся в системе после времени тестирования τ, равно:



где I -- общее число машинных команд, которое предполагается постоянным в рамках этапа тестирования.

Предполагается, что значение функции количества ошибок Z(t) пропорционально числу ошибок, оставшихся в программе после израсходованного на тестирование времени τ.

Z (t) = C \* ε2 (τ),

где С -- некоторая постоянная, t - время работы программы без отказов.

Тогда, если время работы программы без отказа t отсчитывается от точки t = 0, а τ остается фиксированным, функция надежности, или вероятность безотказной работы на интервале от 0 до t, равна



1. Статистическая модель Миллса: суть, порядок расчета показателей?

Использование этой модели предполагает внесение в программу некоторого количества известных ошибок. Они вносятся случайным образом и фиксируются в протоколе искусственных ошибок. Специалист, проводящий тестирование, не знает ни количества, ни характера внесенных ошибок до момента оценки показателей надежности по модели Миллса. Предполагается, что все ошибки (как имеющиеся в исследуемой программе, так и искусственно внесенные) имеют равную вероятность быть найденными в процессе тестирования. Тестируя программу в течение некоторого времени, собирают статистику об ошибках. В момент оценки надежности по протоколу искусственных ошибок все ошибки делятся на ошибки собственно программы и искусственные.

Статистическая модель Миллса позволяет оценить не только количество ошибок до начала тестирования, но и степень отлаженности программ.

Для оценки количества ошибок в программе до начала тестирования используется выражение:

 (4)

где: W - количество преднамеренно внесенных в программу ошибок до начала тестирования;

V - количество обнаруженных в процессе тестирования ошибок из числа преднамеренно внесенных;

S - количество «собственных» ошибок программы, обнаруженных в процессе тестирования.

1. Как оценить степень отлаженности программ, используя статистическую модель Миллса?

Если продолжать тестирование до тех пор, пока все ошибки из числа преднамеренно внесенных не будут обнаружены, степень отлаженности программы С можно оценить с помощью выражения



где: S и W = V (равенство значений W и V в данном случае имеет место, поскольку считается, что все преднамеренно внесенные ошибки обнаружены) имеют тот же смысл, что и в предыдущем выражении (4),

*r* - верхний предел (максимум) предполагаемого количества «собственных» ошибок в программе.

Выражения (4) и (5) представляют собой статистическую модель Миллса. Необходимо заметить, что если тестирование будет закончено преждевременно (т. е. раньше, чем будут обнаружены все преднамеренно внесенные ошибки), то вместо выражения (5) следует использовать более сложное комбинаторное выражение (6). Если обнаружено только V ошибок из W преднамеренно внесенных, используется выражение: