МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ   
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

ЦИФРОВАЯ КАФЕДРА

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  | Б.В. Давидович |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ПРАКТИЧЕСКОЙ РАБОТЕ №2 |
| МЕТОДЫ ТЕСТИРОВАНИЯ |
| по курсу: Инженер по тестированию |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. | 1110М |  |  |  | М.П. Корельский |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2022

1. **Цель работы**

В ходе выполнения практической работы необходимо:

* разработать алгоритм решения задачи;
* реализовать индивидуальный вариант задания на любом известном студенту языке программирования. Внести в программу несколько ошибок.
* выполнить тестирование методами белого и черного ящика;
* результаты тестирования представить в виде таблиц.

1. **Выполнение работы**

**2.1 Стратегия белого ящика**

Вариант №5: в одномерном динамическом массиве, состоящем из n элементов, вычислить среднегеометрическое значение ненулевых элементов массива.

Код разработанной программы и программы с ошибками представлен в таблице 1, алгоритм выполнения программ на рисунках 1 и 2. Результаты выполнения при n = 3 и   
A=[1, 2, 3] на рисунках 3 и 4.

Таблица 1 – Код разработанных программ

|  |  |
| --- | --- |
| Исходный код на языке Python | Код  для тестирования с ошибками |
| n = int(input('Введите количество элементов n\n'))  A = list(map(int, input('Введите n элементов массива через пробел\n').split()[:n]))  print(\*A)  def mg ():  #точка a  count = 0  #точка b  S = 1  #точка c  for i in range (**len(A)**):  #точка d  if A[i] != 0:  #точка f  count += 1  #точка g  S \*= A[i]  #точка e  #точка h  if count != 0:  #точка j  mg = pow(S,(1/count))  else:  #точка i  mg = 0  #точка k  return mg  print(mg()) | n = int(input('Введите количество элементов n\n'))  A = list(map(int, input('Введите n элементов массива через пробел\n').split()[:n]))  print(\*A)  def mg ():  #точка a  count = 0  #точка b  S = 1  #точка c  for i in range (**len(A)-1**):  #точка d  if A[i] != 0:  #точка f  count += 1  #точка g  S \*= A[i]  #точка e  #точка h  if count != 0:  #точка j  mg = pow(S,(1/count))  else:  #точка i  mg = 0  #точка k  return mg  print(mg()) |

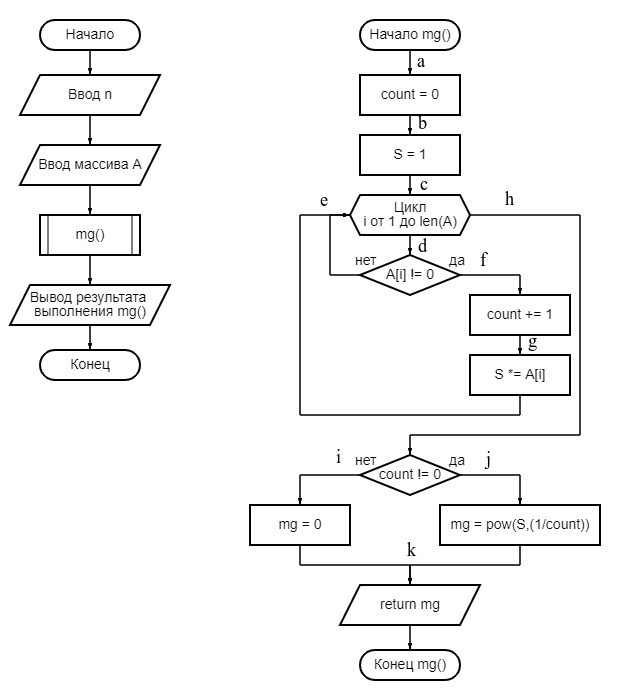


Рисунок 1 – Блок-схема программы

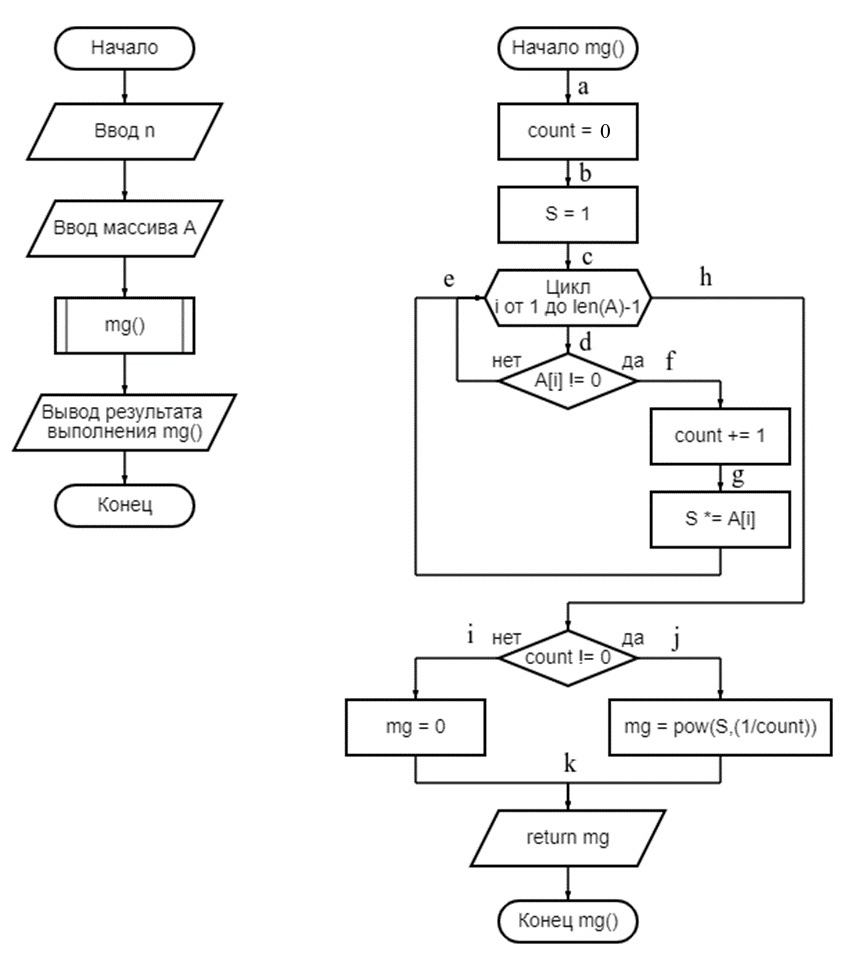


Рисунок 2 – Блок-схема программы с ошибками

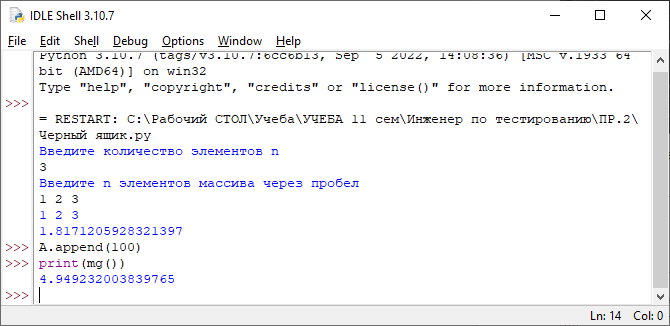


Рисунок 3 – Результат выполнения программы

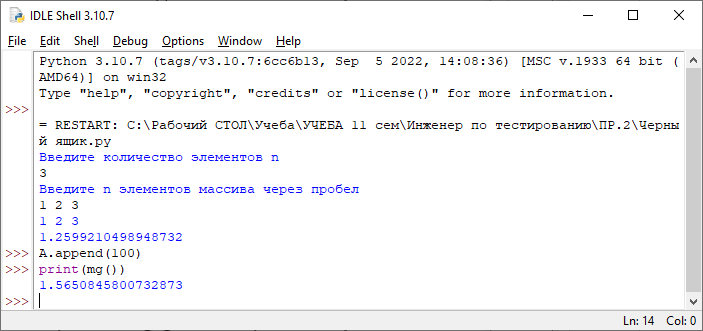


Рисунок 4 – Результат выполнения программы с ошибками

Результаты тестирования **методом покрытия операторов** представлены в таблице 2. Будет реализован путь **abcdfghjk**.

Таблица 2 – Результаты тестирования **методом покрытия операторов**

|  |  |  |  |
| --- | --- | --- | --- |
| Тест | Ожидаемый результат | Фактический результат | Результат тестирования |
| n = 3, A = [1, 2, 3] | mg = 1,817 | mg = 1,414 | Неуспешно |

**По результатам теста видно, что** какой-то оператор работает некорректно.

**Достоинства**: выполнение каждого оператора программы хотя бы один раз.

**Недостатки**: не проверяются невозможные сочетания условий (ошибки могут приводить к возникновению сочетания «невозможных» условий); не проверяется корректность задания локальных или глобальных переменных и вводимых пользователем данных (условия или решения могут работать правильно, но исходные данные для их выполнения могут поступать некорректные); проверяются не все возможные комбинации результатов условия в каждом решении.

Результаты тестирования **методом покрытия решений** представлены в таблице 3. Покрытие всех решений может быть выполнено двумя путями: **abcdfghjk** и **abcdehik**.

Таблица 3 – Результаты тестирования **методом покрытия решений**

|  |  |  |  |
| --- | --- | --- | --- |
| Тест | Ожидаемый результат | Фактический результат | Результат тестирования |
| n = 3, A = [3, 5, 7] | mg = 4,717 | mg = 3,872 | Неуспешно |
| n = 3, A = [0, 0, 0] | mg = 0 | mg = 0 | Успешно |

**По результатам теста видно, что** некорректно работает путь abcdfghjk.

**Достоинства**: каждое направление перехода реализовано, по крайней мере, один раз.

**Недостатки**: не проверяются невозможные сочетания условий (ошибки могут приводить к возникновению сочетания «невозможных» условий); не проверяется корректность задания локальных или глобальных переменных и вводимых пользователем данных (условия или решения могут работать правильно, но исходные данные для их выполнения могут поступать некорректные); проверяются не все возможные комбинации результатов условия в каждом решении.

Результаты тестирования **методом покрытия условий** представлены в таблице 4. В исходном коде имеем 2 условия: {A[i] != 0}, { count != 0}. Требуется достаточное число тестов, чтобы реализовать ситуации, где A[i] != 0, A[i] = 0, count != 0, count = 0. Покрытие всех условий может быть выполнено двумя путями: **abcd(e)fghjk** и **abcdehik**.

Таблица 4 – Результаты тестирования **методом покрытия условий**

|  |  |  |  |
| --- | --- | --- | --- |
| Тест | Ожидаемый результат | Фактический результат | Результат тестирования |
| n = 3, A = [1, 7, 0] | mg = 2,645 | mg = 2,645 | Успешно |
| n = 3, A = [0, 0, 0] | mg = 0 | mg = 0 | Успешно |

**По результатам тестирования видно, что** ошибка не выявляется, когда за цикл выполняется каждый путь abcdfghjk и abcdehjk.

**Достоинства**: все возможные результаты каждого условия в решении выполняются, по крайней мере, один раз.

**Недостатки**: не проверяются невозможные сочетания условий (ошибки могут приводить к возникновению сочетания «невозможных» условий); не проверяется корректность задания локальных или глобальных переменных и вводимых пользователем данных (условия или решения могут работать правильно, но исходные данные для их выполнения могут поступать некорректные); проверяются не все возможные комбинации результатов условия в каждом решении.

Результаты тестирования с использованием **критерия покрытия решений/условий** представлены в таблице 5. Покрытие всех решений/условий может быть выполнено двумя путями: **abcd(e)fghjk** и **abcdehik**.

Таблица 5 – Результаты тестирования **критерием покрытия решений/условий**

|  |  |  |  |
| --- | --- | --- | --- |
| Тест | Ожидаемый результат | Фактический результат | Результат тестирования |
| n = 3, A = [3, 0, 2] | mg = 2,449 | mg = 3 | Неуспешно |
| n = 3, A = [0, 0, 0] | mg = 0 | mg = 0 | Успешно |

**По результатам тестирования видно, что** ошибка выявляется, когда за цикл выполняется каждый путь abcdfghjk и abcdehjk, причем если крайним числом идет 0, то ошибка не выявляется, а если не 0, то выявляется.

**Достоинства**: все возможные результаты каждого условия в решении выполняются по крайней мере один раз, все результаты каждого решения выполняются по крайней мере один раз и каждой точке входа передается управление по крайней мере один раз.

**Недостатки**: не проверяются невозможные сочетания условий (ошибки могут приводить к возникновению сочетания «невозможных» условий); не проверяется корректность задания локальных или глобальных переменных и вводимых пользователем данных (условия или решения могут работать правильно, но исходные данные для их выполнения могут поступать некорректные); проверяются не все возможные комбинации результатов условия в каждом решении. Нужно учитывать, что циклы чувствительны к крайним значениям диапазона перебора.

Результаты тестирования **методом комбинаторного покрытия условий** представлены в таблице 6.

Тесты должны обеспечивать покрытие следующих комбинаций:

1) A[i] = 0, count = 0;

2) A[i] != 0, count != 0.

Покрытие может быть выполнено двумя тестами: **abcd(e)fghjk** и **abcdehik**.

Таблица 6 – Результаты тестирования **методом комбинаторного покрытия условий**

|  |  |  |  |
| --- | --- | --- | --- |
| Тест | Ожидаемый результат | Фактический результат | Результат тестирования |
| n = 3, A = [3, 0, 2] | mg = 2,449 | mg = 3 | Неуспешно |
| n = 3, A = [0, 0, 0] | mg = 0 | mg = 0 | Успешно |

**По результатам тестирования видно, что** ошибка выявляется, когда за цикл выполняется каждый путь abcdfghjk и abcdehjk, причем если крайним числом идет 0, то ошибка не выявляется, а если не 0, то выявляется.

**Достоинства**: проверяются все возможные комбинации результатов условия в каждом решении.

**Недостатки**: не проверяются невозможные сочетания условий; не проверяется корректность задания локальных или глобальных переменных и вводимых пользователем данных (условия или решения могут работать правильно, но исходные данные для их выполнения могут поступать некорректные). Нужно учитывать, что циклы чувствительны к крайним значениям диапазона перебора.

**2.2 Стратегия черного ящика**

Вариант №5: пользователь вводит число. Разработать программу, которая определяет, что он ввел год, а также рассматривает и определяет тип года (високосный или нет).

Описание программы: Пользователь вводит целое число. Годом считается целое число меньше 9999, но большее 0, високосным годом – год, кратный 4, но не кратный 100, а также год, кратный 400. В зависимости от выполнения указанных условий программа выводит «Введен не год», «Введен високосный год» или «Введен невисокосный год».

Код разработанной программы и программы с ошибками представлен в таблице 7. Скриншоты работы исходного кода и кода с ошибками представлены на рисунках 5 и 6 соответственно.

Таблица 7 – Код разработанных программ

| Исходный код на языке Python | Код  для тестирования с ошибками |
| --- | --- |
| A = int(input('Введите число\n'))  if A > 9999 or A **<** 0:  print('Введен не год')  else:  if (A%4 == 0 and A%100 != 0) **or** A%400 == 0:  print('Введен високосный год')  else:  print('Введен невисокосный год') | A = int(input('Введите число\n'))  if A > 9999 or A **<=** 0:  print('Введен не год')  else:  if (A%4 == 0 and A%100 != 0) **and** A%400 == 0:  print('Введен високосный год')  else:  print('Введен невисокосный год') |

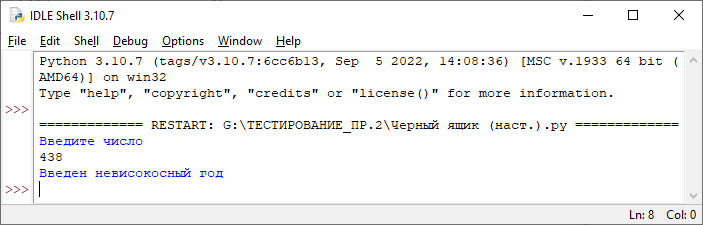


Рисунок 5 – Результат выполнения программы

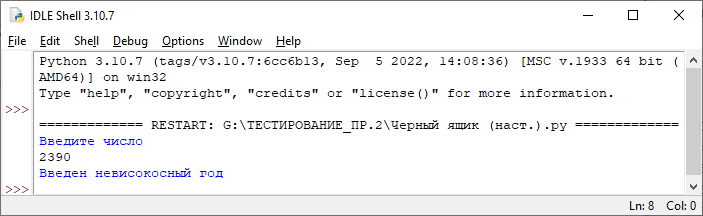


Рисунок 6 – Результат выполнения программы с ошибками

Результаты тестирования **методом эквивалентного разбиения** представлены в таблице 8. Классы эквивалентности исходя из описания программы могут быть следующими: «не год», «високосный год» и «невисокосный год».

Так как «високосный год» определяется по нескольким условиям, а про «невисокосный год» ничего не сказано, то класс «високосный год» разобьем на подклассы: «год, кратный 4, но не кратный 100» и «год, кратный 400». Класс «не год» описывается непересекающимися областями, значит целесообразно включить по одному значению из каждой области, а также дополнительно проверить «не целые числа», т.е. любой тип данных кроме int.

Для результатов теста введем обозначения: нг – не год, вг – високосный год, нвг – невисокосный год.

Таблица 8 – Результаты тестирования **методом эквивалентного разбиения**

|  |  |  |  |
| --- | --- | --- | --- |
| Тест | Ожидаемый результат | Фактический результат | Результат тестирования |
| A = 32,5 | Ошибка | Ошибка | Успешно |
| A = -100 | нг | нг | Успешно |
| A = 10400 | нг | нг | Успешно |
| А = 4 | вг | нвг | Неуспешно |
| A = 800 | вг | нвг | Неуспешно |
| А = 333 | нвг | нвг | Успешно |

**По результатам тестирования видно, что** ошибка в определении класса високосный год.

**Достоинства**: проверяются все классы эквивалентности.

**Недостатки**: нужна высокая компетентность тестировщика, чтобы обеспечить выделение всех возможных классов эквивалентности.

Результаты тестирования **методом граничных значений** представлены в таблице 9.

Таблица 9 – Результаты тестирования **методом граничных значений**

|  |  |  |  |
| --- | --- | --- | --- |
| Тест | Ожидаемый результат | Фактический результат | Результат тестирования |
| A = 32,5 | Ошибка | Ошибка | Успешно |
| A = -1, 0 | нг, вг | нг, нг | Неуспешно |
| A = 9999, 10000 | нвг, нг | нвг, нг | Успешно |
| А = 3, 4, 5 | нвг, вг, нвг | нвг, нвг, нвг | Неуспешно |
| A = 799, 800, 801 | нвг, вг, нвг | нвг, нвг, нвг | Неуспешно |

**По результатам тестирования видно, что** некорректно определяется класс високосный год и граница класса «не год».

**Достоинства**: проверяются все классы эквивалентности и их граничные условия.

**Недостатки**: нужна высокая компетентность тестировщика, чтобы обеспечить выделение всех возможных классов эквивалентности.

Для выполнения **анализа причинно-следственных связей** определим множество входных условий:

1) для определения год/не год – «A – целое число», «A < 0» и «A > 9999»;

2) для определения високосный/невисокосный год – «A кратно 4», «A не кратно 100» и «A кратно 400».

Выходные условия: «введен не год», «введен невисокосный год», «введен високосный год».

Теперь построим таблицу истинности для входных условий. Результаты представлены в таблице 10. Для невозможных сочетаний условий тесты не разрабатывались.

Таблица 10 – Таблица истинности условий

| A = int | A < 0 | A > 9999 | A%4=0 | A%100!=0 | A%400=0 | Результат |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | Х | Х | Х | Х | Х | Ошибка |
| 1 | 1 | 0 | Х | Х | Х | нг |
| 1 | 0 | 1 | Х | Х | Х | нг |
| 1 | 0 | 0 | 0 | 0 | 0 | Невозможное сочетание |
| 1 | 0 | 0 | 1 | 0 | 0 | нвг |
| 1 | 0 | 0 | 0 | 1 | 0 | нвг |
| 1 | 0 | 0 | 1 | 1 | 0 | вг |
| 1 | 0 | 0 | 0 | 0 | 1 | Невозможное сочетание |
| 1 | 0 | 0 | 1 | 0 | 1 | вг |
| 1 | 0 | 0 | 0 | 1 | 1 | Невозможное сочетание |
| 1 | 0 | 0 | 1 | 1 | 1 | Невозможное сочетание |

Результаты тестирования **методом анализа причинно-следственных связей** представлены в таблице 11.

Таблица 11 – Результаты тестирования **методом анализа причинно-следственных св.**

|  |  |  |  |
| --- | --- | --- | --- |
| Тест | Ожидаемый результат | Фактический результат | Результат тестирования |
| А = 32,5 | Ошибка | Ошибка | Успешно |
| А = -100 | нг | нг | Успешно |
| А = 10302 | нг | нг | Успешно |
| А = 100 | нвг | нвг | Успешно |
| А = 3 | нвг | нвг | Успешно |
| А = 4 | вг | нвг | Неуспешно |
| А = 800 | вг | нвг | Неуспешно |

**По результатам тестирования видно, что** некорректно определяется класс високосный год, но невисокосный год и не год определяются корректно. Ошибка должна заключаться в связи условий A%4=0, A%100!=0 и A%400=0. Ошибка в граничных условиях не выявляется.

**Достоинства**: можно выявить ошибки в причинно-следственных связях и в зависимости от корректности и полноты спецификации их детализировать.

**Недостатки**: не проверяются границы выполнения условий.

Результаты тестирования **методом предположения об ошибке** представлены в таблице 12. Ошибка может возникнуть при введении числа с 0 дробной частью, т.е. типа float (согласно описанию, пользователь вводит целое число и подразумевается тип int). Также лучше проверить граничные значения для каждого из совместно проверяемых связанных условий (совместно проверяются условия А%4=0 и А%100!=0).

Таблица 12 – Результаты тестирования **методом предположения об ошибке**

|  |  |  |  |
| --- | --- | --- | --- |
| Тест | Ожидаемый результат | Фактический результат | Результат тестирования |
| A = 40,0 | Ошибка | Ошибка | Успешно |
| A = -1, 0 | нг, вг | нг, нг | Неуспешно |
| A = 9999, 10000 | нвг, нг | нвг, нг | Успешно |
| А = 99, 100, 101 | нвг, нвг, нвг | нвг, нвг, нвг | Успешно |
| А = 3, 4, 5 | нвг, вг, нвг | нвг, нвг, нвг | Неуспешно |
| A = 399, 400, 401 | нвг, вг, нвг | нвг, нвг, нвг | Неуспешно |

**По результатам тестирования видно, что** некорректно определяются класс високосный год и граница класса «не год», но невисокосный год определяется корректно. Ошибка должна заключаться в связи условий (A%4=0, A%100!=0) и A%400=0. С помощью предположения об ошибке мы выявили, что условия A%4=0 и A%100!=0 работают вместе корректно, но при добавлении условия A%400=0 или невыполнении условия A%100!=0 возникает ошибка. Некорректно определяется граница класс «не год».

**Достоинства**: при комбинировании нескольких методов тестирования и высоком уровне экспертности можно выявить «скрытые» ошибки, а также увеличить глубину определения ошибок.

**Недостатки**: требуется опытный тестировщик.

**Выводы**

В ходе выполнения работы проведено тестирования разработанных программ методами белого и черного ящиков. Результаты тестирования, недостатки и достоинства методов отражены по тексту. По полученным результатам можно сделать вывод о том, что наибольшую эффективность дает совместное применение нескольких методов, т.к. при их применении по отдельности из-за своей методической ограниченности некоторые виды ошибок не могут быть выявлены.