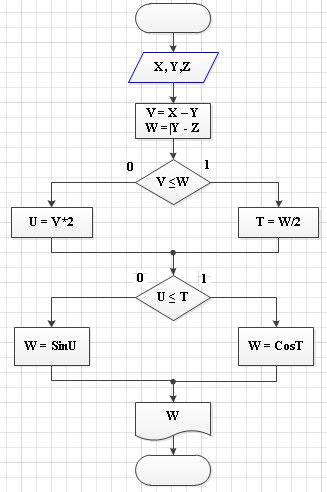
Часть А



Название if 1

Сравнить числа

Даны x,y,z

U=0;t=0

Формула: v=x-y, w=y-z

Далее идет сравнение V и W

Если v<=w то t=w/2

Иначе

u=v\*2

🡺следующее условие if 2

Если u<=t то w=Math.Cos(t)

Иначе w=Math.Sin(u)

Console.WriteLine(“w={0}”,w);

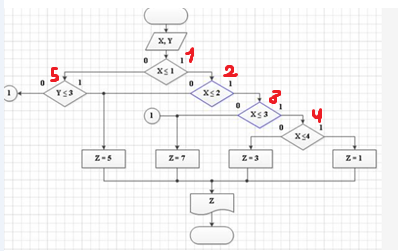
1) x=4;y=3;z=2

V=1;w=1

Тестирование

|  |  |  |  |
| --- | --- | --- | --- |
| **Тест-case** | **Ожидаемый результат** | **Фактический результат** | **Результат тестирования** |
| **1** | 0,5 | 0,5 | Тест пройден |
| **2** | 0,8775825619 | 0,8775825619 | Тест пройден |

Часть Б



Название If 1

Даны x,y

🡺Сравниваем

X<=1 если да то сравниваем со след условием

Иначе сравниваем со след условием

Название If 2

X<=2 если да то сравниваем со след условием

Иначе z=5

Название If 3

X<=3

То сравниваем со след условием

Иначе

Z=7

Название if 4

X<=4 если да то z=1

Иначе z=3

Название if 5

y<=3 если да то z=5

иначе вылет программы

вывод z

1) x=5,2 y=2

2)х=1 у=2

3) x=r y=2

4)x=2 y=1

5) x=6 y=3

|  |  |  |  |
| --- | --- | --- | --- |
| **Тест-case** | **Ожидаемый результат** | **Фактический результат** | **Результат тестирования** |
| **1** | 5 | ошибка | Тест не пройден |
| **2** | 1 | 1 | Тест пройден |
| **3** | Введите число | ошибка | Тест не пройден |
| **4** | 5 | 5 | Тест пройден |
| **5** | 5 | 5 | Тест пройден |

|  |  |  |  |
| --- | --- | --- | --- |
| **Тест-case c ошибкой** | **Фактический результат** | **Причины** | **Вывод** |
| 1 | Ошибка | Из-за того что не переведен в тип double | Отправить приложение на доработку |
| 3 | Ошибка | Т.к в программе нужно ввести только числа, а не какие то другие символы | Отправить приложение на доработку |

Контрольные вопросы

1) Логическая ошибка - это, пожалуй, наиболее серьезная из всех ошибок. Когда написанная программа на любом языке компилирует и работает правильно, но выдает неправильный вывод, недостаток заключается в логике основного программирования. Это ошибка, которая была унаследована от недостатка в базовом алгоритме. Сама логика, на которой базируется вся программа, является ущербной. Чтобы найти решение такой ошибки нужно фундаментальное изменение алгоритма. Вам нужно начать копать в алгоритмическом уровне, чтобы сузить область поиска такой ошибки.

2) Синтаксическая ошибка -каждый компьютерный язык, такой как C, Java, Perl и Python имеет специфический синтаксис, в котором будет написан код. Когда программист не придерживаться "грамматики" спецификациями компьютерного языка, возникнет ошибка синтаксиса. Такого рода ошибки легко устраняются на этапе компиляции.

3) Ошибка компиляции -Компиляция это процесс, в котором программа, написанная на языке высокого уровня, преобразуется в машиночитаемую форму. Многие виды ошибок могут происходить на этом этапе, в том числе и синтаксические ошибки. Иногда, синтаксис исходного кода может быть безупречным, но ошибка компиляции все же может произойти. Это может быть связано с проблемами в самом компиляторе. Эти ошибки исправляются на стадии разработки.

4) Ошибки среды выполнения (RunTime) - Программный код успешно скомпилирован, и исполняемый файл был создан. Вы можете вздохнуть с облегчением и запустить программу, чтобы проверить ее работу. Ошибки при выполнении программы могут возникнуть в результате аварии или нехватки ресурсов носителя. Разработчик должен был предвидеть реальные условия развертывания программы. Это можно исправить, вернувшись к стадии кодирования.

5)Арифметическая ошибка -Многие программы используют числовые переменные, и алгоритм может включать несколько математических вычислений. Арифметические ошибки возникают, когда компьютер не может справиться с проблемами, такими как "Деление на ноль", или ведущие к бесконечному результату. Это снова логическая ошибка, которая может быть исправлена  только путем изменения алгоритма.

6) Ошибки ресурса -Ошибка ресурса возникает, когда значение переменной переполняет максимально допустимое значение. Переполнение буфера, использование неинициализированной переменной, нарушение прав доступа и переполнение стека - примеры некоторых распространенных ошибок.

7)Ошибка взаимодействия -Они могут возникнуть в связи с несоответствием программного обеспечения с аппаратным интерфейсом или интерфейсом прикладного программирования. В случае веб-приложений, ошибка интерфейса может быть результатом неправильного использования веб-протокола.

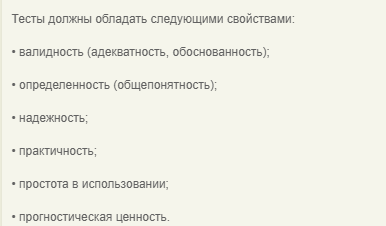
2)   
**Тест** — это процедура, которая позволяет либо подтвердить, либо опровергнуть работоспособность кода.

Обладает универсальностью,точностью и др

3)

**Тестирование чёрного ящика** или **поведенческое тестирование** — стратегия (метод) тестирования функционального поведения объекта (программы, системы) с точки зрения внешнего мира, при котором не используется знание о внутреннем устройстве (коде) тестируемого объекта. Иначе говоря, тестированием чёрного ящика занимаются тестировщики, не имеющие доступ к исходному коду приложения.

4)



5)

