Приложение 1

Ожидаемое описание ошибок

Описание ошибок, обнаруженных студентами, подается в форме текстового документа (формата Word, HTML, TXT или другого общедоступного) и должно содержать следующую информацию.

1. ФИО обнаружившего ошибку
2. Дата и время обнаружения (не обязательно совсем точные)
3. Краткое описание ошибки в виде одного предложения (например, «При некоторых значениях аргументов метод XXX создает исключение YYY»)
4. Указание на нарушаемое ошибкой требование (или требования, описывающие поведение в той ситуации, в которой ошибка происходит)
5. По возможности точное и полное описание условий возникновения ошибки и ее проявлений, которое удается дать, вместе с описанием требуемого поведения (с учетом возможного недетерминизма и трудности выявления всех условий, например, «Если первый аргумент метода XXX является степенью двойки, а второй — отрицателен, метод XXX создает исключение YYY с сообщением “ZZZ”примерно в 70% случаев, хотя в соответствии с требованиями RRR и SSS исключений в такой ситуации возникать не должно, а должен возвращаться результат TTT»)
6. Код как можно более краткого теста, наиболее явно демонстрирующего ошибку (при недетерминизме следует выбирать возможно не самый краткий тест, при котором вероятность проявления ошибки как можно больше)
7. Исправление которое нужно предпринять чтобы исправить ошибку

Приложение 2

Инспекция кода метода возведения в степень

Интерфейс: метод **`int pow(int a, int b)`** в классе **`root.pow.Power`**
Требования:

1. Предусловие тривиально, т.е., метод должен работать для всех целочисленных значений своих параметров
2. В качестве результата метод возвращает результат возведения первого аргумента в степень, равную второму, со следующими уточнениями
    a. При нулевом значении второго аргумента и любом значении первого должен возвращаться результат 1
    b. При отрицательных значениях второго аргумента и любом значении первого должен возвращаться результат 1 (т.е. отрицательный второй аргумент приравнивается к 0).
    c. При переполнении (т.е., если точный результат возведения в степень превосходит по абсолютной величине 2^{31}) возвращается результат возведения в степень по модулю 2^{32}.

Описание реализации.

Метод **`int pow(int a, int b)`** класса **`root.pow.Power`** реализует дихотомический алгоритм быстрого возведения в степень. Перед проведением инспекции нужно ознакомится с описанием алгоритма.
Дихотомический алгоритм быстрого возведения целого числа *a* в степень *b* состоит в следующем.
Степень *b* представим в двоичной записи
$b = (b_k b_{k-1} \dots b_1 b_0)_2 = 2^k b_k + 2^{k-1} b_{k-1} + \dots + 2^1 b_1 + 2^0 b_0$
Тогда $a^b = a^{2^k b_k + 2^{k-1} b_{k-1} + \cdots + 2 b_1 + b_0} = a \quad = ((\dots((a^{b_k})^2 a^{b_{k-1}})^2 \dots)^2 a^{b_1})^2 a^{b_0}$
Будем вычислять последовательно a_i при i=0..k и r_i при i=-1..k так, что
$r_{-1} = 1$
$r_0 = a^{b_0}$ $a_0 = a$
$r_1 = (a^{b_1})^2 a^{b_0}$ $a_1 = a^2$
$r_2 = ((a^{b_2})^2 a^{b_1})^2 a^{b_0}$ $a_2 = a^4$
…
$r_k = ((\dots((a^{b_k})^2 a^{b_{k-1}})^2 \dots)^2 a^{b_1})^2 a^{b_0}$ $a_k = a^{2^k}$
При этом получается, что можно последовательно вычислять $a_{i+1} = a_i^2$ и $r_{i+1} = r_i$ при $b_{i+1} = 0$ или $r_{i+1} = r_i a_{i+1}$ при $b_{i+1} = 1$. В итоге r_k дает нужный результат.

Требуется соотнести представленный алгоритм с кодом метода **`int pow(int a, int b)`** класса **`root.pow.Power`** и либо убедиться, что код работает так, как предписывается алгоритмом (при некоторой разумной интерпретации используемых в методе переменных), либо выявить расхождения.