

## Homework 2

### Фролов Иван Григорьевич, БПИ-235

#### Ошибка 1: Неверная инициализация и логика

1. ФИО обнаружившего ошибку: Фролов Иван Григорьевич
2. Дата и время обнаружения: 30.01.2026 16:00
3. Краткое описание ошибки в виде одного предложения: При любых ненулевых значениях аргументов метод `pow` возвращает неверный результат 0, из-за неправильной инициализации переменной `r` и неверной логики алгоритма.
4. Указание на нарушающее ошибкой требование: Нарушаются требования 2а, 2б Приложения 2.
5. По возможности точное и полное описание условий возникновения ошибки и ее проявлений: Например, вызов `pow(2, 3)` инициализирует `r = 0`. В первой итерации `r` остается 0, затем `r *= r` дает 0. В итоге метод всегда возвращает 0.
6. Код как можно более краткого теста, наиболее явно демонстрирующего ошибку:

```
public static void main(String[] args) {  
    Power p = new Power();  
    System.out.println("pow(2, 3) = " + p.pow(2, 3)); // Ожидается: 8,  
Получено: 0  
}
```

7. Исправление, которое нужно предпринять, чтобы исправить ошибку: Переменную `r` необходимо инициализировать единицей.

### Ошибка 2: Недостаточная проверка и обработка переполнения

1. ФИО обнаружившего ошибку: Фролов Иван Григорьевич
2. Дата и время обнаружения: 30.01.2026 16:10
3. Краткое описание ошибки в виде одного предложения: Метод `pow` не делает проверку на переполнение.
4. Указание на нарушающее ошибкой требование: 2с Приложения 2.
5. По возможности точное и полное описание условий возникновения ошибки и ее проявлений: Например, `pow(2, 31)` или `pow(2, 30)`. Точный результат может превысить `Integer.MAX_VALUE`. В Java происходит "заворачивание" (wraparound) результата, что эквивалентно операции по модулю  $(2^{32})$ . Однако спецификация требует взятия по модулю 2 в степени 31.
6. Код как можно более краткого теста, наиболее явно демонстрирующего ошибку:

```
public static void main(String[] args) {
    Power p = new Power();
    System.out.println("pow(2, 31) = " + p.pow(2, 31));
}
```

7. Исправление, которое нужно предпринять, чтобы исправить ошибку: переписать код, используя взятие остатка по модулю 2 в степени 31.

## Ошибка 3: Неверная реализация дихотомического алгоритма и полное исправление кода

1. ФИО обнаружившего ошибку: Фролов Иван Григорьевич
2. Дата и время обнаружения: 30.01.2026 16:15
3. Краткое описание ошибки в виде одного предложения: Логика внутри цикла не соответствует дихотомическому алгоритму возведения в степень (Exponentiation by Squaring), описанному в Приложении 2, из-за неверной последовательности операций  $r *= a$  и  $r *= r$ .
4. Указание на нарушающее требование: Нарушается описание реализации алгоритма в Приложении 2.
5. По возможности точное и полное описание условий возникновения ошибки и ее проявлений: При положительном  $b$  и  $a > 1$ , выполнение  $r *= r$  после  $\text{if}((b&1) != 0) r *= a;$  приводит к тому, что промежуточный результат  $r$  возводится в квадрат вместо переменной, содержащей текущее основание  $a$ . Например,  $\text{pow}(2, 3)$  возвращает 64 вместо 8.
6. Код как можно более краткого теста, наиболее явно демонстрирующего ошибку:

```
public static void main(String[] args) {
    Power p = new Power();
    System.out.println("pow(2, 3) = " + p.pow(2, 3)); // Ожидается: 8,
Получено: 64
}
```

7. Исправление, которое нужно предпринять, чтобы исправить ошибку: Полностью переписать логику цикла, используя отдельную переменную для основания (`base`) и меняя порядок операций, чтобы соответствовать корректному алгоритму Exponentiation by Squaring, а также добавить явную обработку отрицательных  $b$ .

```
package root.pow;

/**
 * @author Victor Kuliamin
 */
public class Power {

    public int pow(int a, int b) {
```

```
if (b < 0) {
    return 1; // Требование 2б
}
if (b == 0) {
    return 1; // Требование 2а
}

long result = 1;
long base = a;
long MOD = (1L << 31); // 2^31

while (b > 0) {
    if ((b & 1) != 0) {
        result = (result * base) % MOD; // Явное взятие модуля 2^31
    }
    base = (base * base) % MOD; // Явное взятие модуля 2^31
    b >>= 1;
}

return (int) result;
}
```