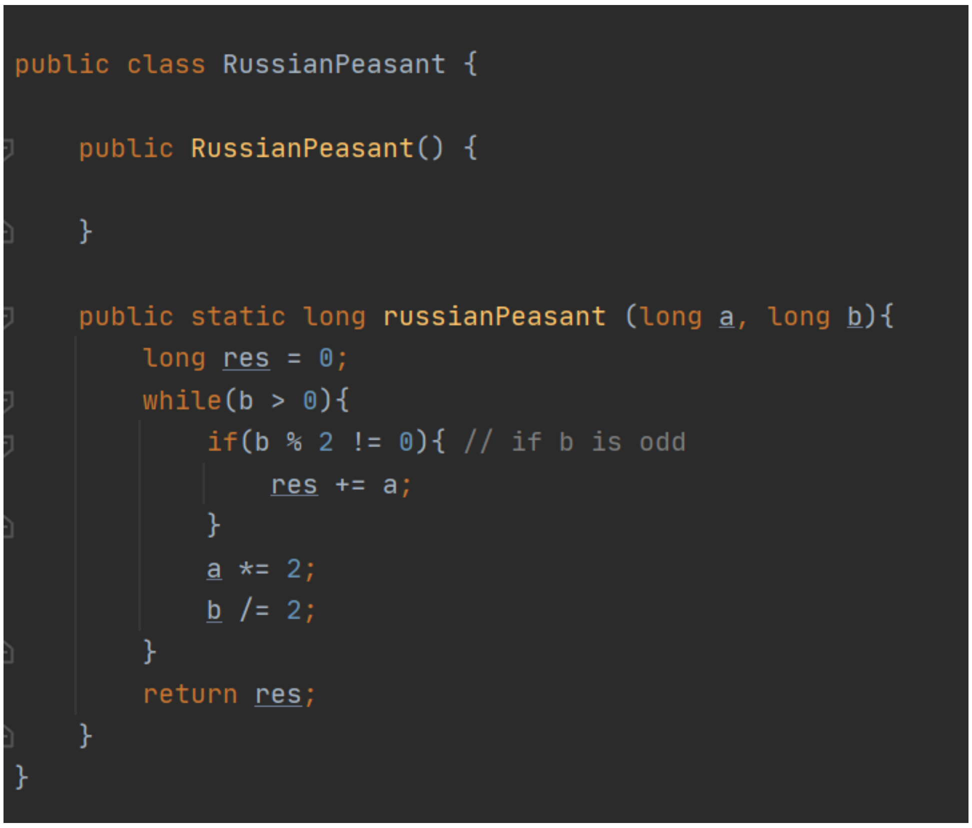
Practical 2: Complexity Analysis

1. Implement the Russian Peasant's algorithm in Java and verify its correctness.



```java
public class RussianPeasant {

    public RussianPeasant() {

    }

    public static long russianPeasant (long a, long b){
        long res = 0;
        while(b > 0){
            if(b % 2 != 0){ // if b is odd
                res += a;
            }
            a *= 2;
            b /= 2;
        }
        return res;
    }
}
```

I choose to approach this question with option B

Option B:
2. Your algorithm should be able to take two integers as arguments, multiply them together and then provide the product.

```java
long[] aa = new long[]{12, 69, 490, 2000, 50000, 342423};
long[] bb = new long[]{5, 86, 230, 5687, 79583, 986975};
System.out.println("Hello - I have choosen Option B for this Analysis");

Stopwatch timer = new Stopwatch();
for (int i = 0; i < bb.length; i++) {

    RussianPeasant.russianPeasant(aa[i], bb[i]);
    StdOut.println("Test " + i + " elapsed time = " + timer.elapsedTime());
    StdOut.println("Result of " + aa[i] + " * " + bb[i] + " = " +
RussianPeasant.russianPeasant(aa[i], bb[i]) + "\n");
}
```

3. Test your algorithm for correctness and then with very large input integers (you may need to use BigInteger for this).

```
StdOut.println("This is testing for BigInts");
Stopwatch timerForBigInt = new Stopwatch();
for (int i = 0; i < bb.length; i++) {

    aa[i] *= 64;
    bb[i] *= 64;
    RussianPeasant.russianPeasant(aa[i], bb[i]);
    StdOut.println("Test " + i + " elapsed time = " +
timerForBigInt.elapsedTime());
    StdOut.println("Result of " + aa[i] + " * " + bb[i] + " = " +
RussianPeasant.russianPeasant(aa[i], bb[i]) + "\n");
}
```

*Instead of utilising BigIntegers I multiplied each argument by 64.

Results from testing my Russian Peasant method

Testing on small Integers

Algorithm	Input	Time (seconds)	Results	Correct?
RussianPeasant	12*5	0.008	60	Y
	69*86	0.033	5934	Y
	490*230	0.033	112700	Y
	2000*5687	0.033	11374000	Y
	50000*79583	0.033	3979150000	Y
	342423*986975	0.033	337962940425	Y

Testing on Big Integers

Algorithm	Input	Time (seconds)	Results	Correct?
RussianPeasant	768 * 320	0.00	245760	Y
	4416 * 5504	0.001	24305664	Y
	31360 * 14720	0.001	461619200	Y
	128000 * 363968	0.001	46587904000	Y
	3200000 * 5093312	0.001	16298598400000	Y
	21915072 * 63166400	0.001	1384296203980800	Y

1. What do you think is the complexity of your algorithm and why?

The time complexity of the Russian Peasant is 0(1). I believ this is the case as there is an upper bound on how long it can take to caluclate results and as we can see in the timing, this upper bound will not be exceeded.