tick1 submission from James Wood

Name	James Wood (jdw74)
College	ROBIN
Submission contents	uk/ac/cam/jdw74/tick1/PackedLong.java uk/ac/cam/jdw74/tick1/SetBit.java uk/ac/cam/jdw74/tick1/TestBit.java uk/ac/cam/jdw74/tick1/answers.txt
Ticker	UNKNOWN
Ticker signature	

PackedLong.java

```
package uk.ac.cam.jdw74.tick1;

public class PackedLong {

    /*
    * Unpack and return the nth bit from the packed number at index position;
    * position counts from zero (representing the least significant bit)
    * up to 63 (representing the most significant bit).
    */
    public static boolean get(long packed, int position) {
        // set "check" to equal 1 if the "position" bit in "packed" is set to 1
        long check = packed >> position & 1L;
        return (check == 1L);
    }

    /*
    * Set the nth bit in the packed number to the value given
    * and return the new packed number
    */
    public static long set(long packed, int position, boolean value) {
        if (value) {
            packed |= 1L << position;
            // update the value "packed" with the bit at "position" set to 1
        }
        else {
            packed &= ~(1L << position);
            // update the value "packed" with the bit a "position" set to 0
        }
        return packed;
    }
}
```

SetBit.java

```
package uk.ac.cam.jdw74.tick1;

public class SetBit {
    public static void main(String [] args) throws Exception {
        long currentValue = Long.decode(args[0]);
        int position = Integer.parseInt(args[1]);
        boolean value = Boolean.parseBoolean(args[2]);
        currentValue = PackedLong.set(currentValue,position,value);
        System.out.println(currentValue);
    }
}
```

TestBit.java

```java
package uk.ac.cam.jdw74.tick1;

public class TestBit {
    public static void main(String[] args) throws Exception {
        long currentValue = Long.decode(args[0]);
        int position = Integer.parseInt(args[1]);
        boolean value = PackedLong.get(currentValue, position);
        System.out.println(value);
    }
}
```

answers.txt

```
- 1.
System.out.println(i = i + 1);
- 2.
    10
    9
On the second line, the expression `j--` sets j to j - 1 and returns the
original value of j (10). j is now 9, and this value is printed on the third
line.
- 3.
    9
    9
- 4.i.
4
- 4.ii.
7
- 5.
One left shift corresponds to multiplication by 2, as numbers are stored in
binary. Therefore, n left shifts corresponds to n multiplications by 2, which
is equivalent to multiplication by 2^n.
- 6.
0x20 & x >> 5  // assuming “nth bit” means “bit with value 2^n”
```