1. The code templates for translating Boolean expressions and if statements are given in Lecture 11. For convenience, they are duplicated as follows:

### Boolean Expressions $E_1 \&\& E_2$:

```plaintext
[[E_1]]
ifeq L1
[[E_2]]
ifeq L1
icnst_1
goto L2
```

L1:
```
icnst_0
```

L2:

### If Statements if (E) then $S_1$ else $S_2$:

```plaintext
[[E]]
ifeq L1
[[S_1]]
goto L2
```

L1:
```
[[S_2]]
```

Consider the following VC code with the local variable indices as indicated.

```plaintext
boolean b1 = true;  // local index 2
boolean b2 = false; // local index 2
int i;            // local index 4
if (b1 && b2)
    i = 0;
else
    i = 1;
```

(a) Translate the above if statement using the code templates provided.

(b) Someone proposes to delete the fourth and fifth instructions from the above code template for Boolean expressions to get the following simpler version:
(c) The Jasmin code produced using the modified code template for Boolean expressions can be further improved. If we know that the boolean expression \( b_1 \& \& b_2 \) is, in fact, the conditional expression for an if statement, there is no need to emit the `ifeq` instruction as indicated in the code template for `if`. The following code is shorter:

```java
.var 2 is b1 Z from L0 to L1
.var 3 is b2 Z from L0 to L1
.var 4 is i I from L0 to L1

.iload_2
.ifeq L5
.iload_3
.ifeq L5
.iconst_0
.istore 4
.goto L6

L5:

.iconst_1
.istore 4

L6:
```

Discuss the code template you will use to generate the above optimised code. Can the same code template be used for translating assignments such as \( b = b_1 \& \& b_2 \)?