Michael Camara
Honor Code Pledge: This work is mine unless otherwise cited
CMPSC 220
Lab 4: Control Structures (Installment 1)

Problem 2)

0	bipush 10	push constant 10 onto the stack
2	istore_1	pop stack and store in location 1 (i = 10)
3	bipush 20	push constant 20 onto the stack
5	istore_2	pop stack and store in location 2 (j = 20)
6	iconst_0	push constant 0 onto the stack
7	istore_3	pop stack and store in location 3 (k = 0)
8	iload_1	push contents of location 1 onto the stack (i (10) → stack)
9	bipush 10	push constant 10 onto the stack)
11	if_icmple 23 (+12)	pop two elements from the stack and compare using the "less-than-or-equal-to" operator, "<=" (if (i <= 10)). If the test is true, immediately go to line 23. This exemplifies short-circuiting in Java, as the next boolean expression in the source code (j == 20) is not even evaluated if this statement is false. This purposefully shortens the amount of code that needs to be evaluated, since the "&&" logical operator requires both boolean expressions in the source code to evaluate as true, and if one is false then it often serves no purpose to evaluate the other.
14	iload_2	push contents of location 2 onto the stack (j (20) → stack)
15	bipush 20	push constant 20 onto the stack
17	if_icmpne 23 (+6)	pop two elements from the stack and compare using the "not-equal-to" operator, "!=" (if (j != 20)). If the test is true, immediately go to line 23. This statement will be skipped entirely if the previous boolean expression in the source code evaluates as false, due to the short circuiting of the "&&" logical operator.
20	bipush 100	push constant 100 onto the stack
22	istore_3	pop stack and store in location 3 (k = 100)
23	iload_3	push contents of location 3 onto stack (k -> stack)
24	ireturn	pop stack and return the integer value (return k)