JVM bytecode instructions

These are selected bytecode instructions, mostly for integers. Note that most of these have variants for other numerical types and references.

iload_x	Loads the integer value of the local variable in slot x on the stack. $x \in \{0, 1, 2, 3\}$
iload X	Loads the value of the local variable pointed to by index X on the top of the stack.
iconst_x	Loads the integer constant x on the stack. x $\in \{0, 1, 2, 3, 4, 5\}$.
istore_x	Stores the current value on top of the stack in the local variable in slot x. $x \in \{0, 1, 2, 3\}$
istore X	Stores the current value on top of the stack in the local variable indexed by X.
ireturn	Method return statement (note that the return value has to have been put on the top of the stack beforehand.
iadd	Pop two (integer) values from the stack, add them and put the result back on the stack.
isub	Pop two (integer) values from the stack, subtract them and put the result back on the stack.
imult	Pop two (integer) values from the stack, multiply them and put the result back on the stack.
idiv	Pop two (integer) values from the stack, divide them and put the result back on the stack.
irem	Pop two (integer) values from the stack, put the result of $x_1\%x_2$ back on the stack.
ineg	Negate the value on the stack.
iinc x, y	Increment the variable in slot x by amount y.
ior	Logical OR for the two integer values on the stack.
iand	Logical AND for the two integer values on the stack.
ixor	Logical XOR for the two integer values on the stack.
ifXX L	Pop one value from the stack, compare it zero according to the operator XX. If the condition is satisfied, jump to the instruction given by label L. XX ∈ { eq, lt, le, ne, gt, ge, null, nonnull }
if_icmpXX L	Pop two values from the stack and compare against each other. Rest as above.
goto L	Unconditional jump to instruction given by the label L.
pop	Discard word currently on top of the stack.
dup	Duplicate word currently on top of the stack.
swap	Swaps the two top values on the stack.
aload_x	Loads an object reference from slot x.
aload X	Loads an object reference from local variable indexed by X.
iaload	Loads onto the stack an integer from an array. The stack must contain the array reference and the index.
iastore	Stores an integer in an array. The stack must contain the arrayreference, the index and the value, in that order.