Questions on Chapters 14 and 15

▼ Chapter 14: Chunks of Bytecode

Multiple Choice Questions (MCQ) - Theoretical Part:

1. **Why does the author mention the quote by Donald Knuth at the beginning of the chapter?**
   - A) To emphasize the importance of theoretical knowledge in programming.
   - B) To highlight the balance between theory and practice in programming.
   - C) To criticize theoretical approaches in programming.
   - D) To promote the idea of spending all time on practical implementation.

   Answer: B) To highlight the balance between theory and practice in programming.

2. **What is the primary reason jlox is deemed insufficient for the book's purpose?**
   - A) It relies on the JVM, limiting understanding of interpreter workings.
   - B) It lacks a proper implementation of memory allocation.
   - C) It uses a tree-walk interpreter, leading to inefficiencies.
   - D) It depends on the C standard library.

   Answer: C) It uses a tree-walk interpreter, leading to inefficiencies.

3. **Why is the tree-walk interpreter considered slow for a general-purpose, imperative language like Lox?**
   - A) It lacks support for high-level, declarative languages.
   - B) It relies on microcode, slowing down execution.
   - C) It has issues with spatial locality and cache efficiency.
   - D) It cannot handle scripting languages efficiently.

   Answer: C) It has issues with spatial locality and cache efficiency.

4. **What is the major disadvantage of the AST walker in terms of memory efficiency?**
    - A) Each syntax piece becomes an AST node, consuming more memory.
    - B) It lacks portability across different platforms.
    - C) It relies on the C standard library for basics.
    - D) It is too dependent on the underlying machine code.

    Answer: A) Each syntax piece becomes an AST node, consuming more memory.

True/False Questions - Practical Part:

1. **True or False: Compiling directly to native code is considered the easiest option for achieving both speed and portability.**
   Answer: False
2. **True or False: Bytecode, despite its emulation layer overhead, provides the advantage of portability, making it suitable for running on various hardware.**
   Answer: True
3. **True or False: The author suggests that the structure of bytecode resembles real machine code, allowing for efficient cache usage.**
   Answer: True
4. **True or False: The term "p-code" in bytecode doesn't stand for "Pascal" but rather "portable" because it was designed for easy adaptation to different architectures.**
   Answer: True

Practical Part - Code Implementation:

1. **Consider the provided Lox script. Which part of the script contributes significantly to the inefficiency of the tree-walk interpreter?**

```
fun fib(n) {
  if (n < 2) return n;
  return fib(n - 1) + fib(n - 2);
```

```
}

var before = clock();
print fib(40);
var after = clock();
print after - before;
```

- A) The clock function
- B) The recursive call in the fib function
- C) The conditional statement
- D) The print statements

Answer: B) The recursive call in the fib function.

2. **What is the primary reason for choosing bytecode over walking the AST or compiling to native code in this context?**
   - A) Better memory efficiency
   - B) Improved spatial locality
   - C) Portability with acceptable performance
   - D) Enhanced support for high-level, declarative languages

   Answer: C) Portability with acceptable performance.

Multiple Choice Questions (MCQs):

1. Regarding the `main` function in `main.c`:

a. What does the `main` function return?

```
1. A non-zero value.
2. The value of `argc`.
3. The value of `const char* argv[]`.
4. 0 (zero).
```

Answer: 4. 0 (zero)

2. In the `common.h` file:

a. What is the purpose of `common.h`?

```
1. To include common C libraries.
2. To define common types and constants.
3. To declare the main function.
4. To initialize dynamic arrays.
```

Answer: 2. To define common types and constants.

3. Concerning the `chunk.h` file:

a. What does `OpCode` represent?

```
1. The name of the source code file.
2. A dynamic array of instructions.
3. The operation code for bytecode instructions.
4. A function to initialize a chunk.
```

Answer: 3. The operation code for bytecode instructions.

4. In the `chunk.h` file:

a. What is the purpose of the `Chunk` struct?

```
1. To declare the main function.
2. To define common types and constants.
3. To represent a dynamic array of instructions.
4. To include common C libraries.
```

Answer: 3. To represent a dynamic array of instructions.

5. Regarding dynamic arrays in `chunk.h`:

a. What are the advantages of dynamic arrays mentioned?

```
1. Constant-time indexed element lookup.
2. Cache-friendly, dense storage.
3. Constant-time appending to the end of the array.
4. All of the above.
```

Answer: 4. All of the above.

True/False Questions:

6. True or False: The `realloc` function in `memory.c` can be used to both allocate new memory and resize existing memory.

Answer: True

7. True or False: The `GROW_ARRAY` macro in `memory.h` is responsible for allocating memory for a dynamic array.

Answer: False

8. True or False: The `freeChunk` function in `chunk.c` deallocates memory used by a `Chunk` and leaves it in an undefined state.

Answer: False

9. True or False: The `initChunk` function in `chunk.c` initializes a new `Chunk` by allocating memory for its dynamic array.

Answer: False

10. True or False: The `writeChunk` function in `chunk.c` is responsible for appending a byte to the end of a `Chunk`'s dynamic array.

Answer: True

Multiple Choice Questions (MCQ):

1. **What is the primary purpose of the disassembler created in the "debug" module?**
    - A. To optimize the bytecode execution.
    - B. To assemble machine code.
    - C. To provide a human-friendly view of the bytecode.
    - D. To debug the interpreter's internal representation.

    Answer: C

2. **In the `disassembleChunk` function, why is the offset not directly incremented in the loop, and what is responsible for incrementing it?**
    - A. Offset is incremented manually for better control flow.
    - B. The loop increments the offset automatically.
    - C. To handle different sizes of instructions.
    - D. To avoid control flow issues in the loop.

    Answer: C

3. **What is the role of the `simpleInstruction` function in the disassembler?**
    - A. To execute the opcode.
    - B. To print the name of the opcode.
    - C. To handle complex instructions.
    - D. To manage memory for instructions.

    Answer: B

True/False Questions:

1. **True or False: The `disassembleInstruction` function reads a single byte from the bytecode to identify the opcode.**
    - **Answer: True**
2. **True or False: The `simpleInstruction` function only prints the opcode name for the OP_RETURN instruction.**

- **Answer: True**

3. **True or False: The disassembler output "0000 OP_RETURN" indicates a successful encoding and decoding of the binary bytecode.**
   - **Answer: True**

Multiple Choice Questions (MCQ):

1. **What type of values are supported in the initial representation of Lox values in the VM?**
   - A. Integers
   - B. Characters
   - C. Double-precision, floating-point numbers
   - D. Strings

   Answer: C

2. **What is the purpose of the `ValueArray` struct in the interpreter?**
   - A. To represent bytecode instructions.
   - B. To manage the dynamic array of constant values.
   - C. To store chunk data.
   - D. To define the structure of constants.

   Answer: B

3. **Why does the interpreter need a dynamic array for values in the constant pool?**
   - A. To efficiently handle immediate instructions.
   - B. To support variable-sized constants like strings.
   - C. To store bytecode instructions.
   - D. To manage alignment and padding.

   Answer: B

True/False Questions:

1. **True or False: The `Value` module is introduced to support integers and characters in addition to double-precision, floating-point numbers.**
   - **Answer: False**
2. **True or False: The `writeValueArray` function in the `Value` module is responsible for inserting values into the constant pool array.**
   - **Answer: True**
3. **True or False: The `addConstant` function in the `Chunk` module returns the index where the constant was appended to the constant array.**
   - **Answer: True**

Multiple Choice Questions (MCQ):

1. **What does the `OP_CONSTANT` opcode represent in the bytecode?**
   - A. It prints a constant value.
   - B. It returns a constant.
   - C. It loads a constant for use.
   - D. It executes a constant operation.

   Answer: C
2. **Why does the `OP_CONSTANT` instruction need an operand in its bytecode format?**
   - A. To store the constant value directly in the code stream.
   - B. To specify the number of constants in the chunk.
   - C. To identify which constant to load from the constant array.
   - D. To provide information about the data type of the constant.

   Answer: C

True/False Questions:

1. **True or False: The `constantInstruction` function in the `debug` module is responsible for printing the constant index in the disassembled output.**
    - **Answer: False**
2. **True or False: The `printValue` function in the `value` module is used to display the constant values in the disassembled output.**
    - **Answer: True**
3. **True or False: The `return offset + 2;` statement in the `constantInstruction` function indicates the offset of the beginning of the next instruction in the bytecode.**
    - **Answer: True**

Multiple Choice Questions (MCQ):

1. **What information is stored in the array parallel to the bytecode array in a `Chunk` to keep track of source line information?**
    - A. Token values
    - B. AST nodes
    - C. Line numbers
    - D. Constant values

    Answer: C
2. **Why does the `Chunk` store line information in a separate array instead of interleaving it in the bytecode?**
    - A. To reduce the memory usage of the bytecode.
    - B. To improve cache efficiency during runtime.
    - C. To speed up the compilation process.
    - D. To enable parallel processing of instructions.

    Answer: B

True/False Questions:

1. **True or False: The `writeChunk` function now takes an additional parameter, `int line`, to specify the source line number of the bytecode.**
    - **Answer: True**
2. **True or False: The `disassembleInstruction` function now displays the source line number for each instruction in the disassembled output.**
    - **Answer: True**
3. **True or False: The reduction of AST classes in `clox` compared to `jlox` is one of the reasons why the new interpreter is expected to be faster.**
    - **Answer: True**

▼ Chapter 15: A Virtual Machine

Multiple Choice Questions (MCQ):

1. **What is the purpose of the global `VM` object declared in the `vm.c` module?**
    - A. To store the bytecode instructions.
    - B. To keep track of multiple virtual machines.
    - C. To handle memory allocation for the VM.
    - D. To avoid passing a VM pointer to various functions.

    Answer: D
2. **Why does the author mention that using a static VM instance might not be a sound engineering choice for a real language implementation?**
    - A. It limits the host application's flexibility.
    - B. It makes the code heavier.
    - C. It leads to global variable-related issues.
    - D. It affects the book's readability.

    Answer: A

True/False Questions:

1. **True or False: The global `VM` object is declared to store the bytecode instructions for execution.**
    - **Answer: False**
2. **True or False: The `initVM` function is called when the interpreter starts, and the `freeVM` function is called before exiting.**
    - **Answer: True**

Multiple Choice Questions (MCQ):

1. **What is the primary purpose of the `run` function in the VM?**
    - A. Handle memory allocation for the VM.
    - B. Interpret and execute bytecode instructions.
    - C. Manage the constant pool in the chunk.
    - D. Implement the disassembly of bytecode.

    Answer: B
2. **What does the `IP` stand for in the context of the VM?**
    - A. Index Pointer
    - B. Instruction Pointer
    - C. Immediate Pointer
    - D. Incremental Pointer

    Answer: B
3. **Why is `READ_CONSTANT()` used as a macro in the `run` function?**
    - A. To define a new constant in the VM.
    - B. To read a constant value from the bytecode.
    - C. To increment the instruction pointer.
    - D. To execute a constant instruction.

    Answer: B

True/False Questions:

1. **True or False: The `DEBUG_TRACE_EXECUTION` flag is used to enable diagnostic logging in the VM.**
   - **Answer: True**
2. **True or False: The `run` function, which is responsible for executing bytecode instructions, is a highly complex component of the VM.**
   - **Answer: False**
3. **True or False: The `READ_BYTE` and `READ_CONSTANT` macros are explicitly undefined at the end of the `run` function to manage their scoping.**
   - **Answer: True**

Multiple Choice Questions (MCQ):

1. **In the Lox expression `print 3 - 2;`, which part of the expression is evaluated first?**
   - A. The subtraction operation
   - B. The constant `3`
   - C. The constant `2`
   - D. The `print` statement

   Answer: B
2. **What is the primary purpose of using a stack to manage temporary values in the VM?**
   - A. To optimize the execution speed of the VM.
   - B. To allow for parallel execution of instructions.
   - C. To handle recursive function calls.
   - D. To manage the order of evaluation of expressions.

   Answer: D
3. **In the diagram showing the execution of the Lox program step by step, what does each bar represent?**

- A. The execution time of an instruction.
- B. A temporary value that needs to be preserved.
- C. The order of appearance of numbers in the program.
- D. The duration of a recursive function call.

Answer: B

True/False Questions:

1. **True or False: The order of appearance of numbers in the Lox program determines their lifetime during execution.**
   - **Answer: True**
2. **True or False: The stack in the VM is used to manage temporary values, and it operates in a Last-In, First-Out (LIFO) manner.**
   - **Answer: True**
3. **True or False: In the second diagram, each time a number is introduced, it is pushed onto the stack from the left.**
   - **Answer: False**

Multiple Choice Questions (MCQ):

1. **What is the primary reason for using a stack-based VM?**
   - A. To optimize execution speed
   - B. To handle recursive function calls
   - C. To simplify the compilation process
   - D. To provide a more intuitive model for programmers

   Answer: C
2. **In the Lox VM's stack, where is the bottom of the stack located?**
   - A. At the first array element (index 0)
   - B. At the last array element (index STACK_MAX - 1)

- C. Just before the first array element
- D. Just after the last array element

Answer: A

3. **How is the stack size limited in the Lox VM?**
   - A. Dynamically grows as needed
   - B. Fixed size defined by STACK_MAX
   - C. Adjusts based on available memory
   - D. Auto-resizes during execution

   Answer: B

True/False Questions:

1. **True or False: The stack pointer points to the last element of the stack.**
   - **Answer: False**
2. **True or False: When pushing a value onto the stack, the stack pointer is incremented to the next available slot after storing the value.**
   - **Answer: True**
3. **True or False: In the `push` function, the `stackTop` pointer points just past the last used element.**
   - **Answer: True**

Fill-in-the-Blank:

1. The maximum number of values that can be stored on the stack in the Lox VM, for now, is defined by the constant _____.

   Answer: STACK_MAX

2. The `pop` function in the Lox VM moves the _____ to get to the most recently used slot in the array.

   Answer: stack pointer

Multiple Choice Questions (MCQ):

1. **What is the purpose of the `OP_NEGATE` bytecode instruction in the Lox VM?**
   - A. To print the negation of a constant value
   - B. To negate the top value on the stack
   - C. To handle unary negation in the source code
   - D. To return the negation of the result

   Answer: B

True/False Questions:

1. **True or False: The `OP_NEGATE` bytecode instruction directly operates on constants.**
   - **Answer: False**
2. **True or False: The `OP_NEGATE` instruction first pops a value from the stack, negates it, and then pushes the result back onto the stack.**
   - **Answer: True**

Fill-in-the-Blank:

1. In the example `print -a;`, the bytecode instruction generated for the unary negation is _____.

   Answer: OP_NEGATE

2. The `OP_NEGATE` instruction is used to negate the _____ value on the stack.

   Answer: top

Multiple Choice Questions (MCQ):

1. **What is the purpose of the `BINARY_OP` macro in the Lox VM?**
   - A. To disassemble binary arithmetic operations
   - B. To evaluate binary arithmetic operations

- C. To generate bytecode for binary arithmetic operations
- D. To negate binary values

Answer: B

2. **In the `BINARY_OP` macro, why is the order of popping operands reversed (assigning the first popped operand to `b`)?**
   - A. It follows the natural order of arithmetic operations
   - B. It is a mistake in the implementation
   - C. It avoids a stack overflow
   - D. It aligns with the order of operand evaluation in expressions

   Answer: D

True/False Questions:

1. **True or False: The `BINARY_OP` macro is used to handle all binary arithmetic operators in the Lox VM, using the specified C operator as an argument.**
   - **Answer: True**
2. **True or False: The disassembler instructions for binary arithmetic operators do not include information about the operands.**
   - **Answer: True**

Fill-in-the-Blank:

1. The `OP_ADD` bytecode instruction is used to perform _____ in the Lox VM.

   Answer: addition

2. In the sequence of instructions for the expression `((1.2 + 3.4) / 5.6)`, the last bytecode instruction is _____.

   Answer: OP_RETURN

Fill-in-the-Blank:

1. The `OP_DIVIDE` bytecode instruction is used to perform _____ in the Lox VM.

   Answer: division

2. The stack in a stack-based VM is often used to shuttle _____ around between different instructions.

   Answer: values

True/False Questions:

1. **True or False: The `OP_NEGATE` bytecode instruction negates the top value on the stack in the Lox VM.**
   - **Answer: True**
2. **True or False: The `BINARY_OP` macro uses a do-while loop to contain multiple statements and allows a semicolon at the end.**

- **Answer: True**

Multiple Choice Questions:

1. **How does the Lox VM handle stack initialization?**
   - A. Allocating a new stack array
   - B. Clearing the unused cells in the array
   - C. Setting the stack pointer to point at the end of the array
   - D. Setting the stack pointer to point at the beginning of the array

   Answer: D