Review questions Chapter 06 **Low-level languages** **(does not include the last two sections, Algorithms and Testing)**

1. True or False? Machine language is the set of binary-coded instructions that are executed directly by a computer.

2. True or False? Each machine language instruction performs a single complex task, such as sorting a list of numbers.

3. True or False? Very few programs are written in machine language today.

4. True or False? A virtual computer is a hypothetical machine in which there are no limits on memory use.

5. True or False? The Pep/8 machine is a virtual computer.

6. True or False? The word length of the Pep/8 machine is 2 bytes.

7. True or False? Unlike a real computer, the Pep/8 machine does not have an instruction register (IR).

8. True or False? All instructions in the Pep/8 machine use the operand specifier.

9. True or False? The Pep/8 instruction specifier contains an addressing mode specifier that indicates how the operand should be interpreted.

10. True or False? The Pep/8 machine language includes an instruction to stop the execution of a program.

11. True or False? The Pep/8 character input instruction specifies the register into which a character is to be stored.

12. True or False? Only direct addressing is allowed with the Pep/8 character input instruction.

13. True or False? The Pep/8 system includes a simulator that can be used to input and run programs.

14. True or False? The loader is software that puts a machine-language program into memory so that it can be executed.

15. True or False? Machine language programs are loaded into the Pep/8 simulator by specifying the instructions in hexadecimal.

16. True or False? Assembly language allows program instructions to be specified using mnemonics that correspond to machine language instructions.

17. True or False? An assembler is used to execute an assembly language program directly on the central processing unit.

18. True or False? An assembler directive is an instruction to the assembler itself.

19. True or False? A comment in a program is explanatory text for the human reader.

20. True or False? An assembler ignores comments.

21. . True or False? In Pep/8 assembly language, you can allocate data storage space of various sizes, give these locations names, and refer to them by name later in the program.

22. True or False? In the Pep/8 machine, the contents of the A register can be compared to the contents of a place in memory.

23. True or False? In Pep/8 assembly language, decisions can be made using instructions that check the status of the accumulator.

24. True or False? Assembly language is an abstraction, hiding some of the details that occur at the machine language level.

25. . Which language is actually executed by the central processing unit of a computer?
A. high-level language
B. assembly language
C. machine language
D. virtual language
E. accumulator language

26. . Which of the following best describes a virtual computer?
A. a hypothetical computer with unlimited memory
B. a hypothetical computer with an unlimited instruction set
C. a hypothetical computer used to illustrate the features of a real machine
D. a programmed simulator for a real CPU like a Pentium 4
E. a programmed simulator of multiple CPUs

27. . Which register contains the address of the next instruction to be executed?
A. program counter
B. instruction register
C. index register
D. accumulator
E. status register

28. . Which register holds a copy of the instruction being executed?
A. program counter
B. instruction register
C. index register
D. accumulator
E. status register

29. . Which register holds the results of operations?
A. program counter
B. instruction register
C. index register

D. accumulator
E. status register

30. . How big is the Pep/8 program counter?
A. 8 bits
B. 16 bits
C. 24 bits
D. 32 bits
E. 64 bits

31. . How big is the Pep/8 instruction register?
A. 8 bits
B. 16 bits
C. 24 bits
D. 32 bits
E. 64 bits

32. . How big is the Pep/8 accumulator?
A. 8 bits
B. 16 bits
C. 24 bits
D. 32 bits
E. 64 bits

33. . How big is each addressable memory location in the Pep/8 machine?
A. 8 bits
B. 16 bits
C. 24 bits
D. 32 bits
E. 64 bits

34. . How big is each memory address in the Pep/8 machine?
A. 8 bits
B. 16 bits
C. 24 bits
D. 32 bits
E. 64 bits

35. . Which part of the Pep/8 instruction specifier indicates which instruction is to be carried out?
A. operation code
B. register specifier
C. addressing mode specifier
D. status bit
E. accumulator

36. . Which part of the Pep/8 instruction specifier indicates how the operand should be interpreted?
A. operation code
B. register specifier
C. addressing mode specifier
D. status bit
E. accumulator

37. Which Pep/8 addressing mode indicates that the operand contains data rather than the location of data?
A. accumulator
B. direct
C. immediate
D. virtual
E. status

38. Which Pep/8 addressing mode indicates that the operand contains the location of data rather than the data itself?
A. accumulator
B. direct
C. immediate
D. virtual
E. status

39. . Which of the following is not an operation that can be performed by a Pep/8 machine instruction?
A. stop execution
B. load the operand into the accumulator
C. store the contents of the accumulator into the operand
D. add the contents of the program counter to the accumulator
E. read character input and store into the operand

40. . What does a loader do?
A. loads a machine language program into memory
B. loads an assembly language program into memory
C. loads the accumulator with zeros
D. loads one instruction into the instruction register
E. loads one operand into memory

41. Which language uses mnemonics to represent instructions?
A. high-level language
B. assembly language
C. machine language
D. virtual language
E. accumulator language

42. . Which of the following is not a valid mnemonic in the Pep/8 assembly language?
A. STOP
B. LOADA
C. ADDA
D. STOREA
E. REPEAT

43. What is an assembler directive?
A. an assembly language instruction
B. an instruction to the assembler program
C. a human readable comment
D. an alternative way to specify the operand

E. an instruction that begins the assembly language translation

44. Which of the following represent the function(s) of the accumulator (A register)?
A. The accumulator holds data ONLY.
B. The accumulator holds the results of the computer operations ONLY.
C. The accumulator holds data AND the results of computer operations.
D. The accumulator holds the address of the next instruction to be executed.
E. The accumulator holds the copy of the instruction currently being executed.

45. Since the word length in the Pep/8 virtual computer is 2 bytes, how long is the information that flows into and out of the arithmetic/logic unit (ALU) of the PEP/8's CPU?
A. 16 bits
B. 8 bits
C. 32 bits
D. 64 bits
E. 128 bits

46. All of the following represent sample subsets of Pep/8 instructions EXCEPT:

A. Opcode 0000 →Instruction: Stop execution
B. Opcode 1100→Instruction: Load the operand into the A register
C. Opcode 1110→Instruction: Add the operand to the A register
D. Opcode 01001→Instruction: Character input to the operand
E. Opcode 10111→Store the contents of the operand into the A register

47. ._______________ instructions are executed directly by the hardware of a particular computer.

48. A _______________ is a hypothetical machine used to illustrate the features of a real machine.

49. The _______________ is a register used to hold the address of the next instruction to be executed.

50. The _______________ is a register used to hold a copy of the instruction being executed.

51. The _______________ is a register used to hold the results of operations.

52. In the Pep/8 machine, the _______________ of the instruction indicates which instruction is to be carried out.

53. In the Pep/8 machine, the _______________ of the instruction indicates how the operand should be interpreted;

54. In Pep/8, the _______________ addressing mode indicates that the operand contains data, rather than the address of data.

55. In Pep/8, the _______________ addressing mode indicates that the operand contains the address of data, rather than the data itself.

56. Instructions that do not have an operand are called _______________ instructions.

57. . The Pep/8 load instruction loads the operand into the _______________.

58. The Pep/8 store instruction stores the contents of the _______________ into the operand.

59. The Pep/8 _______________ allows a program to be loaded and executed.

60. Pep/8 machine language program instructions are entered using _______________ digits.

61. A _______________ is a program that puts a machine-language program into memory so that it can be executed.

62. An assembly-language program uses _______________ to represent instructions.

63. An _______________ is a program that translates an assembly-language program into machine code.

64. The input to an assembler is an _______________ program.

65. The output of an assembler is a _______________ program.

66. An _______________ is an instruction for the translating program.

67. A _______________ is explanatory text added to a program for the benefit of the human reader.

68. In Pep/8 the instructions that test the contents of the accumulator are used to make _______________ in a program.

69. The _______________ and the ________________ are the two parts to the instruction in the Pep/8 virtual machine.

70. _________________________ are instructions to the translating program.

71. During the process of running a program in an assembly language , the ________________ to the assembler is a program written in an assembly language; while, the _________________from the assembler is a program written in machine code.

72. A ______________________ is defined as a section of code that repeats.

73. What is a machine-language instruction?

74. What is the significance of the relationship between a machine-language instruction and the binary string that represents it?

75. What is a virtual computer?

76. Why are most programs written in high-level languages?

77. How many hexadecimal digits are needed to describe the bit pattern in a byte? Why?

78. What is the word length of the Pep/8 machine?

79. What is a register?

80. Name three registers used in the Pep/8 machine.

81. What is a memory address?

82. What are the two main parts of a Pep/8 instruction?

83. Name the three parts of the instruction specifier.

84. What interpretation is applied to the operand if the addressing mode is direct?

85. . What interpretation is applied to the operand if the addressing mode is immediate?

86. In a Pep/8 instruction, if the addressing mode is direct, the leftmost four bits of the operand specifier are not used. Explain.

87. Why are the three rightmost bits of the instruction specifier ignored in the case of the Stop instruction?

88. In what sense can an instruction (such as Load) be interpreted two different ways?

89. Given the following state of memory, show the contents of the accumulator after the execution of this Load instruction:

```
11000000 00000000 00000010

0001  A2
0002   11
0003   FF
```

90. Given the following state of memory, show the contents of the accumulator after the execution of

this Load instruction:

11000001 00000000 00000010

0001 A2
0002 00

0003 11

91. Given the following state of memory, show the contents of the accumulator after the execution of this Load instruction:

11000001 00000000 00000010

0001 A2
0002 FF
0003 11

92. Given the following state of memory, show the contents of the accumulator after the execution of this Load instruction:

11000001 00000000 00000011

0001 A2
0002 00
0003 11

Answer: Unable to determine. The first byte of the accumulator is 00010001 and the second byte is whatever is stored in memory location 0004.

93. .Given the following state of memory, show the contents of the accumulator after the execution of the following two instructions (the first operation is Load, the second is Add):

11000000 00000000 00000001
01110000 00000000 00000001

0001 A2
0002 00
0003 FE

Answer: 10100010 00000001

94. Given the following state of memory, show the contents of the accumulator after the execution of the following two instructions (the first operation is Load, the second is Add):

11100001 00000000 00000001
01110000 00000000 00010001

0001 A2
0002 11
0003 FE

Answer: 10100010 00100010

95. If the input character is X, what is the result of executing the following two instructions (the first operation is Character Input, the second is Character Output):

```
0001    01001001 00000000 00000110
0004    01010000 00000000 00001010
```

Answer: The character X is written to the screen (the first instruction overwrites the operand of the second instruction, which uses immediate addressing).

96. Write the Pep/8 instruction that loads the contents of location 0002 into the accumulator (the opcode for Load is 1100).

97. Write the Pep/8 instruction that loads the value 15 into the accumulator (the opcode for Load is 1100).

98. Write the Pep/8 instruction that inputs a character from the keyboard and stores it into memory location 0003 (the opcode for Character Input is 01001).

99. Write the Pep/8 instruction that inputs a character from the keyboard and stores it into its own operation specifier (the opcode for Character Input is 01001).

100. What is an assembler directive?

101. .What does the following assembler directive do?

```
.ASCII  "Hello\x00"
```

102. What does the following assembler directive do?

```
.BLOCK 2
```

103. What does the following assembler directive do?

```
.WORD 4
```

104. What does an assembler accept as input and what does it produce as output?

105. Assuming the following assembly language program starts at location 0000, at what memory location is the first “l” of the string “Hello” stored?

```
CHARO 0x0048,i  ;Output ‘H’
CHARO 0x0065,i  ;Output ‘e’
CHARO 0c006C,i  ;Output ‘l’
CHARO 0x006C,i  ;Output ‘l’
CHARO 0x006Fi  ;Output ‘o’
STOP
.END
```

106. Write an assembly language program that implements the following algorithm.

```
Read num1
Read num2
Read num3
Load num1
Add num3
Sub num2
Store in answer
Write answer
```

107. What is a virtual computer (machine) and what is its purpose?

108. Where is the operand if the address mode specifier is 001?

109. Where is the operand if the address mode specifier is 000?

110. Why is the distinction between the immediate addressing mode and the direct addressing mode important?

111. Describe the two levels of programming in Pep/8.

112. Discuss the abstraction provided by assembly language programming.

113. How can a concrete step in one language be an abstract step in another language?

114. . Compare and contrast a virtual computer and an actual computer.

115. . Defend or attack the common practice of software piracy engaged in by millions of computer users throughout the world.

116. Define a Comment and its purpose in programming.

Solutions and answers

1. True or False? Machine language is the set of binary-coded instructions that are executed directly by a computer.

Answer: True

2. True or False? Each machine language instruction performs a single complex task, such as sorting a list of numbers.

Answer: False

3. True or False? Very few programs are written in machine language today.

Answer: True

4. True or False? A virtual computer is a hypothetical machine in which there are no limits on memory use.

Answer: False

5. True or False? The Pep/8 machine is a virtual computer.

Answer: True

6. True or False? The word length of the Pep/8 machine is 2 bytes.

Answer: True

7. True or False? Unlike a real computer, the Pep/8 machine does not have an instruction register (IR).

Answer: False

8. True or False? All instructions in the Pep/8 machine use the operand specifier.

Answer: False

9. True or False? The Pep/8 instruction specifier contains an addressing mode specifier that indicates how the operand should be interpreted.

Answer: True

10. True or False? The Pep/8 machine language includes an instruction to stop the execution of a program.

Answer: True

11. True or False? The Pep/8 character input instruction specifies the register into which a character is to be stored.

Answer: False

12. True or False? Only direct addressing is allowed with the Pep/8 character input instruction.

Answer: True

13. True or False? The Pep/8 system includes a simulator that can be used to input and run programs.

Answer: True

14. True or False? The loader is software that puts a machine-language program into memory so that it can be executed.

Answer: True

15. True or False? Machine language programs are loaded into the Pep/8 simulator by specifying the instructions in hexadecimal.

Answer: True

16. True or False? Assembly language allows program instructions to be specified using mnemonics

that correspond to machine language instructions.

Answer: True

17. True or False? An assembler is used to execute an assembly language program directly on the central processing unit.

Answer: False

18. True or False? An assembler directive is an instruction to the assembler itself.

Answer: True

19. True or False? A comment in a program is explanatory text for the human reader.

Answer: True

20. True or False? An assembler ignores comments.

Answer: True

21. . True or False? In Pep/8 assembly language, you can allocate data storage space of various sizes, give these locations names, and refer to them by name later in the program.

Answer: True

22. True or False? In the Pep/8 machine, the contents of the A register can be compared to the contents of a place in memory.

Answer: True

23. True or False? In Pep/8 assembly language, decisions can be made using instructions that check the status of the accumulator.

Answer: True

24. True or False? Assembly language is an abstraction, hiding some of the details that occur at the machine language level.

Answer: True

25. . Which language is actually executed by the central processing unit of a computer?
A. high-level language
B. assembly language
C. machine language
D. virtual language
E. accumulator language

Answer: C

26. . Which of the following best describes a virtual computer?
A. a hypothetical computer with unlimited memory
B. a hypothetical computer with an unlimited instruction set
C. a hypothetical computer used to illustrate the features of a real machine
D. a programmed simulator for a real CPU like a Pentium 4
E. a programmed simulator of multiple CPUs

Answer: C

27. . Which register contains the address of the next instruction to be executed?
A. program counter
B. instruction register
C. index register
D. accumulator
E. status register

Answer: A

28. . Which register holds a copy of the instruction being executed?
A. program counter
B. instruction register
C. index register
D. accumulator
E. status register

Answer: B

29. . Which register holds the results of operations?
A. program counter
B. instruction register
C. index register
D. accumulator
E. status register

Answer: D

30. . How big is the Pep/8 program counter?
A. 8 bits
B. 16 bits
C. 24 bits
D. 32 bits
E. 64 bits

Answer: B

31. . How big is the Pep/8 instruction register?
A. 8 bits
B. 16 bits
C. 24 bits
D. 32 bits
E. 64 bits

Answer: C

32. . How big is the Pep/8 accumulator?
A. 8 bits
B. 16 bits
C. 24 bits
D. 32 bits
E. 64 bits

Answer: C

33. . How big is each addressable memory location in the Pep/8 machine?
A. 8 bits
B. 16 bits
C. 24 bits
D. 32 bits
E. 64 bits

Answer: A

34. . How big is each memory address in the Pep/8 machine?
A. 8 bits
B. 16 bits
C. 24 bits
D. 32 bits
E. 64 bits

Answer: B

35. . Which part of the Pep/8 instruction specifier indicates which instruction is to be carried out?
A. operation code
B. register specifier
C. addressing mode specifier
D. status bit
E. accumulator

Answer: A

36. . Which part of the Pep/8 instruction specifier indicates how the operand should be interpreted?
A. operation code
B. register specifier
C. addressing mode specifier
D. status bit
E. accumulator

Answer: C

37. Which Pep/8 addressing mode indicates that the operand contains data rather than the location of data?
A. accumulator
B. direct
C. immediate
D. virtual
E. status

Answer: C

38. Which Pep/8 addressing mode indicates that the operand contains the location of data rather than

the data itself?
A. accumulator
B. direct
C. immediate
D. virtual
E. status

Answer: B

39. . Which of the following is not an operation that can be performed by a Pep/8 machine instruction?
A. stop execution
B. load the operand into the accumulator
C. store the contents of the accumulator into the operand
D. add the contents of the program counter to the accumulator
E. read character input and store into the operand

Answer: D

40. . What does a loader do?
A. loads a machine language program into memory
B. loads an assembly language program into memory
C. loads the accumulator with zeros
D. loads one instruction into the instruction register
E. loads one operand into memory

Answer: A

41. Which language uses mnemonics to represent instructions?
A. high-level language
B. assembly language
C. machine language
D. virtual language
E. accumulator language

Answer: B

42. . Which of the following is not a valid mnemonic in the Pep/8 assembly language?
A. STOP
B. LOADA
C. ADDA
D. STOREA
E. REPEAT

Answer: E

43. What is an assembler directive?

A. an assembly language instruction
B. an instruction to the assembler program
C. a human readable comment
D. an alternative way to specify the operand
E. an instruction that begins the assembly language translation

Answer: B

44. Which of the following represent the function(s) of the accumulator (A register)?

A. The accumulator holds data ONLY.
B. The accumulator holds the results of the computer operations ONLY.
C. The accumulator holds data AND the results of computer operations.
D. The accumulator holds the address of the next instruction to be executed.
E. The accumulator holds the copy of the instruction currently being executed.

Answer: C

45. Since the word length in the Pep/8 virtual computer is 2 bytes, how long is the information that flows into and out of the arithmetic/logic unit (ALU) of the PEP/8's CPU?

A. 16 bits
B. 8 bits
C. 32 bits
D. 64 bits
E. 128 bits

Answer: A

46. All of the following represent sample subsets of Pep/8 instructions EXCEPT:

A. Opcode 0000 →Instruction: Stop execution
B. Opcode 1100→Instruction: Load the operand into the A register
C. Opcode 1110→Instruction: Add the operand to the A register
D. Opcode 01001→Instruction: Character input to the operand
E. Opcode 10111→Store the contents of the operand into the A register

Answer: E

47. ._______________ instructions are executed directly by the hardware of a particular computer.

Answer: machine language

48. A ________________ is a hypothetical machine used to illustrate the features of a real machine.

Answer: virtual computer

49. The ________________ is a register used to hold the address of the next instruction to be executed.

Answer: program counter

50. The ________________ is a register used to hold a copy of the instruction being executed.

Answer: instruction register

51. The ________________ is a register used to hold the results of operations.

Answer: accumulator (or A register)

52. In the Pep/8 machine, the ________________ of the instruction indicates which instruction is to be carried out.

Answer: operation code (or opcode)

53. In the Pep/8 machine, the ________________ of the instruction indicates how the operand should be interpreted;

Answer: addressing mode specifier

54. In Pep/8, the ________________ addressing mode indicates that the operand contains data, rather than the address of data.

Answer: immediate

55. In Pep/8, the ________________ addressing mode indicates that the operand contains the address

of data, rather than the data itself.

Answer: direct

56. Instructions that do not have an operand are called _______________ instructions.

Answer: unary

57. . The Pep/8 load instruction loads the operand into the _______________.

Answer: accumulator (or A register)

58. The Pep/8 store instruction stores the contents of the _______________ into the operand.

Answer: accumulator (or A register)

59. The Pep/8 _______________ allows a program to be loaded and executed.

Answer: simulator

60. Pep/8 machine language program instructions are entered using _______________ digits.

Answer: Hexadecimal

61. A _______________ is a program that puts a machine-language program into memory so that it can be executed.

Answer: Loader

62. An assembly-language program uses _______________ to represent instructions.

Answer: Mnemonics

63. An _______________ is a program that translates an assembly-language program into machine code.

Answer: Assembler

64. The input to an assembler is an _______________ program.

Answer: assembly language

65. The output of an assembler is a _______________ program.

Answer: machine language

66. An _______________ is an instruction for the translating program.

Answer: assembler directive

67. A _______________ is explanatory text added to a program for the benefit of the human reader.

Answer: comment

68. In Pep/8 the instructions that test the contents of the accumulator are used to make _______________ in a program.

Answer: decisions (or branches)

69. The _______________ and the ________________ are the two parts to the instruction in the Pep/8 virtual machine.

Answer: instruction specifier , 16-bit operand specifier

70. _________________________ are instructions to the translating program.

Answer: Assembler directives

71. During the process of running a program in an assembly language , the ________________ to the assembler is a program written in an assembly language; while, the _________________from the assembler is a program written in machine code.

Answer: input; output (in that exact order)

72. A ______________________ is defined as a section of code that repeats.

Answer: loop

73. What is a machine-language instruction?

Answer: An instruction in a form that can be directly executed by the circuitry of a particular type of CPU.

74. What is the significance of the relationship between a machine-language instruction and the binary string that represents it?

Answer: There is no significance. The actual bit pattern of the code has no special meaning other than it has been assigned to a particular instruction.

75. What is a virtual computer?

Answer: A hypothetical machine designed to illustrate important features of a real machine.

76. Why are most programs written in high-level languages?

Answer: High-level languages can express a complex task in a single instruction that might correspond to many low-level instructions. Thus programming in a high-level language is more efficient.

77. How many hexadecimal digits are needed to describe the bit pattern in a byte? Why?

Answer: Two hex digits represent a byte because each hex digit corresponds to four bits.

78. What is the word length of the Pep/8 machine?

Answer: 2 bytes (or 16 bits)

79. What is a register?

Answer: A small area of storage in the arithmetic/logic unit of the CPU, used to hold special data and intermediate values

80. Name three registers used in the Pep/8 machine.

Answer: The program counter (PC), the instruction register (IR), and the accumulator (A Register).

81. What is a memory address?

Answer: A numeric label or "name" given to a particular location.

82. What are the two main parts of a Pep/8 instruction?

Answer: The instruction specifier and the operand specifier.

83. Name the three parts of the instruction specifier.

Answer: The operation code, the register specifier, and the addressing mode specifier.

84. What interpretation is applied to the operand if the addressing mode is direct?

Answer: The operand holds a memory address from which data will be retrieved or to which data will be stored, depending on the instruction.

85. . What interpretation is applied to the operand if the addressing mode is immediate?

Answer: The operand holds the data itself. That is, the data to be used is part of the instruction.

86. In a Pep/8 instruction, if the addressing mode is direct, the leftmost four bits of the operand specifier are not used. Explain.

Answer: If the addressing mode is direct, the operand specifier contains a 12-bit memory address. Thus the four leftmost bits of the 16-bit operand specifier are not needed.

87. Why are the three rightmost bits of the instruction specifier ignored in the case of the Stop instruction?

Answer: The Stop instruction does not refer to a register or data, so the bits that specify the register and the addressing mode are not used.

88. In what sense can an instruction (such as Load) be interpreted two different ways?

Answer: Depending on the addressing mode, the Load instruction may load the accumulator with the value in the operand specifier, or it may load the accumulator with a value retrieved from memory.

89. Given the following state of memory, show the contents of the accumulator after the execution of this Load instruction:

```
11000000 00000000 00000010

0001  A2
0002   11
0003   FF
```

Answer: 00000000 00000010

90. Given the following state of memory, show the contents of the accumulator after the execution of this Load instruction:

```
11000001 00000000 00000010

0001  A2
0002  00
0003   11
```

Answer: 00000000 00010001

91. Given the following state of memory, show the contents of the accumulator after the execution of this Load instruction:

```
11000001 00000000 00000010

0001  A2
0002   FF
0003   11
```

Answer: 11111111 00010001

92. Given the following state of memory, show the contents of the accumulator after the execution of this Load instruction:

```
11000001 00000000 00000011
```

```
0001  A2
0002   00
0003   11
```

Answer: Unable to determine. The first byte of the accumulator is 00010001 and the second byte is whatever is stored in memory location 0004.

93. .Given the following state of memory, show the contents of the accumulator after the execution of the following two instructions (the first operation is Load, the second is Add):

```
11000000 00000000 00000001
01110000 00000000 00000001
```

```
0001  A2
0002   00
0003   FE
```

Answer: 10100010 00000001

94. Given the following state of memory, show the contents of the accumulator after the execution of the following two instructions (the first operation is Load, the second is Add):

```
11100001 00000000 00000001
01110000 00000000 00010001
```

```
0001  A2
0002   11
0003   FE
```

Answer: 10100010 00100010

95. If the input character is X, what is the result of executing the following two instructions (the first operation is Character Input, the second is Character Output):

```
0001    01001001 00000000 00000110
0004    01010000 00000000 00001010
```

Answer: The character X is written to the screen (the first instruction overwrites the operand of the second instruction, which uses immediate addressing).

96. Write the Pep/8 instruction that loads the contents of location 0002 into the accumulator (the opcode for Load is 1100).

Answer: 11000001 00000000 00000010

97. Write the Pep/8 instruction that loads the value 15 into the accumulator (the opcode for Load is

1100).

Answer: 11000000 00000000 00001111

98. Write the Pep/8 instruction that inputs a character from the keyboard and stores it into memory location 0003 (the opcode for Character Input is 01001).

Answer: 010011001 00000000 00000011

99. Write the Pep/8 instruction that inputs a character from the keyboard and stores it into its own operation specifier (the opcode for Character Input is 01001).

Answer: The Character Input instruction can only be used with direct addressing.

100. What is an assembler directive?

Answer: Instructions to the assembler (as opposed to instructions that the assembler is to translate).

101. .What does the following assembler directive do?

```
.ASCII  "Hello\x00"
```

Answer: It stores the character string "Hello" in memory.

102. What does the following assembler directive do?

```
.BLOCK 2
```

Answer: It generates two bytes of storage.

103. What does the following assembler directive do?

```
.WORD 4
```

Answer: It generates a word of storage and stores the decimal value 4 into it.

104. What does an assembler accept as input and what does it produce as output?

Answer: An assembler takes a program written in assembly language and produces the corresponding program expressed in machine code.

105. Assuming the following assembly language program starts at location 0000, at what memory location is the first “l” of the string “Hello” stored?

```
CHARO 0x0048,i  ;Output ‘H’
CHARO 0x0065,i  ;Output ‘e’
CHARO 0c006C,i  ;Output ‘l’
CHARO 0x006C,i  ;Output ‘l’
CHARO 0x006Fi  ;Output ‘o’
STOP
.END
```

Answer: Memory location 0008.

106. Write an assembly language program that implements the following algorithm.

```
Read num1
Read num2
Read num3
Load num1
Add num3
Sub num2
Store in answer
Write answer
```

Answer:

```
                BR              Main
answer:         .WORD            0x0000
num1:            .BLOCK         2
num2:  .BLOCK         2
num3:            .BLOCK         2

Main:           DECI            num1,d
                DECI             num2,d
                DECI            num3,d
                LDA             num1,d
                ADDA  num3,d
                SUBA            num2,d
                STA             answer,d
                DECO             answer,d
STOP
.END
```

107. What is a virtual computer (machine) and what is its purpose?

Answer: A virtual computer (machine) is a hypothetical machine designed to illustrate key features of real computer. A virtual computer (machine) is established in an operating system’s memory to emulate all the functions of a working computer to permit different software programs to run off the

Central Processing Unit (CPU) at different times.

108. Where is the operand if the address mode specifier is 001?

Answer: The operand is in the place named in the operand specifier.

109. Where is the operand if the address mode specifier is 000?

Answer: The operand is in the operand specifier.

110. Why is the distinction between the immediate addressing mode and the direct addressing mode important?

Answer: The distinction between the immediate addressing mode and the direct addressing mode is extremely important because it determines where the data involved in the operation is stored or will be stored.

111. . Describe the two levels of programming in Pep/8.

Answer: Pep/8 allows the user to program instructions in machine language and in assembly language.

112. Discuss the abstraction provided by assembly language programming.

Answer: Assembly language programming allows the programmer to specify instructions using English-like mnemonics and associated operands. These instructions correspond to the lower-level machine language instructions, but are easier to enter, read, and debug. The details of the binary machine-language code are masked by the easier assembly language code. The cost of this abstraction is that the assembly code must be translated into machine language in order to be executed. But the benefits of working in a language that is easier to deal with far outweigh the extra processing step needed to translate the code.

113. How can a concrete step in one language be an abstract step in another language?

Answer: Languages range from machine language in which every step must be explicitly defined using the machine's own instructions to high-level languages in which single instructions exist for complex processes. Suppose a step in a program design called for a complex calculation to be made. When programming in a high-level language, this step can be written as a single instruction, and thus is a concrete step in that language. To accomplish that same calculation in assembly language would require many separate instructions, and thus could be considered an abstract step that needed further refinement.

114. . Compare and contrast a virtual computer and an actual computer.

Answer: A virtual computer, like the Pep/8 machine, simulates the processing of a real computer. Both can be programmed in a form of machine language and assembly language code, but the virtual computer is really just a program that processes those instructions and produces a result. The machine language of an actual computer is executed through the circuitry of the CPU. A virtual computer simplifies some of the details involved and creates a nice environment for exploring low-level computing issues without getting too bogged down in the details of a real CPU's machine language. Thus, a virtual computer is another example of an abstraction.

115. . Defend or attack the common practice of software piracy engaged in by millions of computer users throughout the world.

Answer: Supporters of upholding the legal integrity of software license agreements argue that research demonstrates that in a single year 107,000 jobs were lost in the United States because of pirated software. These defenders of copyrighted software argue that "softlifting," or duplicating software from an acquaintance's copy represents a form of high-tech economic theft that should be prohibited and prosecuted vigorously. Moreover, these advocates of enforcing copyrighted software license agreements point to the increased risk of exposure of potential computer viruses to the user of pirated software and to those friends who also use the same pirated copy of software.

Opponents of copyrighted software argue that the functionality of software distinguishes itself from other types of intellectual property and makes the need for copyrighting software problematic for computer programmers and users. For example, advocates of open-source code argue that vigorously and strictly enforcing copyrights for computer software restrains its development and improvement. In addition, these advocates assert that mandating licensing fees means certain copyrighted software makes this software too costly for many low income people to afford. They believe that a program's original source code should be in the public domain and available for anyone to download, rewrite, and improve without fear of criminal prosecution for legal copyright infringement.

116. Define a Comment and its purpose in programming.

A Comment is explanatory text written for the human reader of the program that explains what is occurring with the program. Comments are an integral and necessary component of writing a program in any programming language. For example, look at the following "Hello" program.

```
CHARO 0x0048, i; Output an 'H'
CHARO 0x0065, i; Output an 'e'
CHARO 0x006C,I; Output an 'l'
CHARO 0x006C,I; Output an 'l'
CHARO 0X006F,I; Output an 'o'
STOP
.END
```

All the text after the semicolon serve as Comments for the reader of the program, and explain to the reader of the “Hello” program the specific output data being produced when the program is executed.