Valencia Community College
COP 1006: Introduction to Programming Concepts
Competency Exam Study Guide

Objectives

A. [10%] **Computer Math**. Convert between number systems including Hexadecimal, Binary, and Decimal. Understand the meaning of byte, kilobyte, megabyte and gigabyte. Understand the term *hertz*, as it applies to a processor speed, and the terms megahertz and gigahertz.

B. [10%] **Arithmetic operators**. Evaluate arithmetic expressions with the operators +, −, *, /, % (modulus) using rules of precedence.

C. [15%] **Relational and Logical Operators**. Identify the veracity (truth) of expressions using the relational operators >, <, >=, <=, EQ (Equal) and NE (Not Equal). Evaluate truth calculations using logical operators AND, OR and NOT. Know the Truth Tables using logical operators.

D. [15%] **Selection Control Structures**. Trace the flow of control in an algorithm using selection control structures IF – THEN and IF – THEN – ELSE. Understand nested selection control structures, i.e., IF – THEN statements embedded in IF – THEN statements. Trace the flow of control in an algorithm using the multi-selection control structure CASE (also called SWITCH).

E. [10%] **Repetition Control Structures**. Trace the flow of control in an algorithm that uses a WHILE loop construct. Understand the exiting of the loop based on the veracity of the condition associated with the control structure. Understand nested repetition control structures.

F. [10%] **Arrays**. Understand one-dimensional and two-dimensional arrays containing data elements. Recognize a repetition control structure that correctly accesses all of the elements of a one-dimensional array.

G. [5%] **Modularization**. Recognize and understand the concept of a hierarchy chart that demonstrates the concepts of modularization, and scalability of software.

H. [5%] **Common algorithms**. Understand common algorithms for searching and sorting. Be able to walk through algorithms that implement searching and sorting.

I. [5%] **Software development lifecycle**. Understand the software development lifecycle that includes requirements, design, code, test, and maintain. Understand the necessity to have quality assurance in software development.

J. [15%] **Problem solving using formal logic**. (Algorithm development.) Create a solution to a problem described in English. Complexity of this problem statement will include the requirement to nest at least one repetition construct with selection constructs. For example, a correct solution will require the placement of an IF – THEN construct within a WHILE construct.

Exam Symbols and Assumptions

Time limit: 1 hour. No calculators, notes, or other aids are permitted. Scrap paper is permitted.

The following symbols and assumptions will be used throughout the competency exam:

Arithmetic Operators:

+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulus

Assignment Operator: = Assignment

Logical Operators:

AND	Logical And
OR	Logical Or
NOT	Logical Not

Relational Operators:

<	Less Than	>	Greater Than
EQ	Equal To	NE	Not Equal To
<=	Less Than or Equal To	>=	Greater Than or Equal To

Flow of Control:

Selection

IF / THEN / ENDIF	delimit single selection structure
IF / THEN / ELSE / ENDIF	delimit dual selection structure
CASE / ENDCASE	delimit multiple selection structure

Repetition

WHILE / ENDWHILE	structure checking condition at the beginning of iteration
DO / WHILE	structure checking condition at the end of each iteration

Input / Output:

PRINT	display on an output device
READ	obtain value from an input device

Modularization:

GOSUB	execute a module
RETURN	end module processing, and return to calling module

Arrays:

ARRAY	All arrays will assume 0-based indexing
ARRAY values [10]	create array named values with 10 elements indexed 0-9
values[i] = 50	access of array element using a variable as index

Assumptions:

- All data will be numeric or string.
- All data must be initialized by the programmer.
- Standard algebraic order of operation (precedence rules) apply to numeric calculations.
- All values on the right-hand side of an assignment operator will be evaluated before the assignment is made to a variable on the left-hand side.

Choose the BEST answer to each question. On the exam, you will put all answers on the answer sheet provided. Use the preceding information to assist you with the meaning of the questions.

A. Computer Math

Convert between number systems including Hexadecimal, Binary, and Decimal. Understand the storage space of byte, kilobyte, megabyte, gigabyte, and terabtye. Understand the term *hertz*, as it applies to a processor speed, and the terms megahertz and gigahertz.

Decimal

Our usual counting system. We use the digits 0 – 9.

places:	10^3	10^2	10^1	10^0
	1000	100	10	1

Binary

The computer's counting system. It uses the digits 0 and 1.
Each digit is one bit ("binary digit").

places:	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	128	64	32	16	8	4	2	1

Hexadecimal

Base-16 counting system where each digit represents 4 bits. It is more human-readable than binary. Hexadecimal ("hex") uses the digits 0 – 9, and the letters A – F, where

A = 10	B = 11	C = 12	D = 13	E = 14	F = 15
1010	1011	1100	1101	1110	1111

A hex number is usually preceeded by a "0x", as in, 0x37.

places:	16^2	16^1	16^0
	256	16	1

1. What is the result of converting the decimal number 37 to binary?

   A) `1010 0101` B) `0010 0101` C) `0111 0101` D) `0101 0101`

2. What is the result of converting the decimal number 92 to binary?

   A) `0101 1000` B) `0101 0101` C) `0011 1100` D) `0101 1100`

3. What is the result of converting the hex number 0x37 to decimal?

   A) `37` B) `0011 0111` C) `55` D) `775`

4. What is the result of converting the hex number 0x2A to decimal?

   A) `42` B) `0010 1010` C) `25` D) `775`

Storage Space

one ASCII character	=	1 byte	=	8 bits
Kb	=	kilobyte	=	1,000 bytes
Mb	=	megabyte	=	1 million bytes
Gb	=	gigabyte	=	1 billion bytes
Tb	=	terabyte	=	trillion bytes

5. A file containing a few pages of text is stored on a hard disk. The file is most likely to require what amount of storage space?

   A) A few Gigabytes B) A few Kilobytes C) A few Megabytes D) A few bytes

Processor Speed

An *instruction* is the smallest unit of work that the CPU can perform. The CPU executes one instruction, such as add, per clock cycle. A processor with a faster clock rate executes more instructions per second than a processor with a slower clock rate. The clock rate is specified in Hertz:

Hertz is the frequency; the number of cycles per second.

Ex: 500 MHz = 500 million cycles/second
= 500 million instructions/second can be executed

6. A desktop PC is for sale in a local computer shop. The sign indicates that it has 700 MHz. This refers to:

   A) a hard disk containing 700 megabytes of storage.
   B) 700 megabytes of Random Access Memory
   C) the processor speed
   D) quality of the sound coming from the sound card

B. Arithmetic operators

Evaluate arithmetic expressions with the operators +, −, *, /, % (modulus) using rules of precedence.

Rules of Precedence

Arithmetic operators take 2 operands. Higher precedence operators are evaluated before lower-precedence operators. Operators of *equal* precedence are evaluated from left to right.

Highest	( )
	*, /, %
Lowest	−, +

Ex: `a = 3 + 10 / 2;`

```
3 + 10 / 2        Division has higher precedence than addition; do first
 3 + 5
```

Modulus is also known as *remainder*.

Ex:

13 % 3 is 1: Take 4 groups of 3 (4*3=12) out of 13, with 1 left over.
4 % 7 is 4: Take 0 groups of 7 (0*7=0) out of 4, with 4 left over.
8 % 4 is 0: Take 2 groups of 4 (2*4=8) out of 8, with 0 left over.

7. What is the value of **num** after the following statement has been executed?

```
num = 30 + 3 * 2
```

A) 36 B) 66 C) 92 D) 180

8. What is the value of **num** after the following statement has been executed?

```
num = 30 % 7
```

A) 0 B) 2 C) 4 D) 210

9. What is the value of **num** after the following statements have been executed?

```
x = 5
y = x + 2
x =  x + 1
num = x + y
```

A) 8 B) 12 C) 13 D) 15

C. Relational and Logical Operators

Identify the veracity (truth) of expressions using the relational operators >, <, >=, <=, EQ (Equal) and NE (Not Equal). Evaluate truth calculations using logical operators AND, OR and NOT. Know the Truth Tables using logical operators.

The relational operators compare the left hand side of the expression with the right hand side, and evaluate to `true` or `false`.

Ex:		
	`3 < 5`	`true` that 3 is less than 5
	`6 <= 6`	`true` that 6 is less than or equal to 6
	`4 NE 6`	`true` that 4 is not equal to 6
	`2 >= 7`	`false` that 2 is greater than or equal to 7
	`5 EQ 5`	`true` that 5 is equal to 5

Often we need to compare two expressions at the same time using `AND` or `OR`. With `AND`, *both* conditions must be `true` for the entire expression to be `true`. With `OR`, either condition needs to be `true` for the entire expression to be `true`.

Truth Table for `AND`

`true`	`AND`	`true`	`true`
`true`	`AND`	`false`	`false`
`false`	`AND`	`true`	`false`
`false`	`AND`	`false`	`false`

Truth Table for `OR`

`true`	`OR`	`true`	`true`
`true`	`OR`	`false`	`true`
`false`	`OR`	`true`	`true`
`false`	`OR`	`false`	`false`

`NOT` (unary negation) returns the opposite. So `NOT true` is `false`; `NOT false` is `true`.

10. Given that `b = 3, c = 6,` what is the value of the expression?

```
b > 5 AND c > 5
```

A) `true` B) `false` C) it depends D) can't know

11. Given that `c = 5, d = 9,` what is the value of the expression?

```
c < 4 OR d > 6
```

A) `true` B) `false` C) it depends D) can't know

D. Selection Control Structures

Trace the flow of control in an algorithm using selection control structures IF – THEN and IF – THEN – ELSE. Understand nested selection control structures, i.e., IF – THEN statements embedded in IF – THEN statements. Trace the flow of control in an algorithm using the multi-selection control structure CASE (also called SWITCH).

a. Single selection

```
IF (condition) THEN
     statement(s) to execute when the condition is true
ENDIF
```

12. What is the value of **num** after the following statements have been executed?

```
x = 5
y = x + 1
IF (x < y) THEN
      x = x + 1
ENDIF
num = x + y
```

A) 11 B) 12 C) 13 D) 15

13. What will be displayed by the following statements?

```
x = 3
y = x * -1
y = y * y
IF (x < y AND y < 0) THEN
      PRINT x
ENDIF
```

A) -6 B) 3 C) 36 D) nothing

b. Dual selection

```
IF (condition) THEN
     statement(s) to execute when the condition is true
ELSE
     statement(s) to execute when the condition is false
ENDIF
```

14. What is the value of **num** after the following statements have been executed?

```
x = 2
y = x * x + 1
IF (y <= 5) THEN
      num = x + y
ELSE
      num = x – y
ENDIF
```

A) -4 B) -3 C) 7 D) 8

15. What will be displayed by the following statements?

```
x = 5
y = x * -1 + 2
IF (x < y AND y > 0) THEN
      PRINT x * x, y * y
ELSE
      PRINT y * y, x * x
ENDIF
```

A) 9 25 B) 25 9 C) 25 25 D) 25 –25

c. Nested selection

```
IF (conditionA) THEN
     statement(s) to execute when conditionA is true
ELSE
     statement(s) to execute when conditionA is false:
     IF (conditionB) THEN
          statement(s) to execute when conditionB is true
     ELSE
          statement(s) to execute when the conditionB is false
     ENDIF
ENDIF
```

16. What is the value of **num** after the following statements have been executed?

```
x = 1
y = 8
z = 5
IF (x < 7) THEN
     IF (y < 5) THEN
           num = 1
      ELSE
           num = 2
      ENDIF
ELSE
      IF (z > 5) THEN
           num = 3
      ELSE
           num = 4
      ENDIF
ENDIF
```

A) 1 B) 2 C) 3 D) 4

d. Multiple selection

Select from a set number of options.

```
CASE variable
     value_1: statement(s) to execute when variable has value_1
     value_2: statement(s) to execute when variable has value_2
     value_3: statement(s) to execute when variable has value_3
     …
ENDCASE
```

17. What will be displayed by the following statements?

```
x = 1
IF (x < 7 AND x*3 > 2) THEN
      num = 1
ELSE
      num = 3
ENDIF

CASE num
   1:       PRINT 4
   2:       PRINT 3
   3:       PRINT 2
   4:       PRINT 1
ENDCASE
```

A) 4 B) 3 C) 2 D) 1

E. Repetition Control Structures

Trace the flow of control in an algorithm that uses a WHILE loop construct. Understand the exiting of the loop based on the veracity of the condition associated with the control structure. Understand nested repetition control structures.

```
WHILE (condition)
      statements
ENDWHILE
```

The `condition` is checked, and if `true`, we enter the loop and execute the statements inside the loop. On reaching the `ENDWHILE`, we re-evaluate the `condition`. If it is still `true`, we execute the statements inside the loop again. If the condition is `false`, the loop is done.

18. How many times will the `PRINT` statement be executed in the following code?

```
outer = 1
WHILE (outer < 2)
      inner = 1
      WHILE (inner <= 2)
            PRINT inner * outer
            inner = inner + 1
      ENDWHILE
      outer = outer + 1
ENDWHILE
```

A) 2 B) 3 C) 5 D) 6

F. Arrays

Understand one-dimensional and two-dimensional arrays containing data elements. Recognize a repetition control structure that correctly accesses all of the elements of a one-dimensional array.

An array is a data structure used to hold a group of related items that have the same data type. They are accessed by the name of the array and index into the array. Indices will begin at 0; in other words, the first item of array `myData` is `myData[0]`. Example usage of one-dimensional array would be to hold grades for a single student, or a shopping cart for a single customer:

`ARRAY cart [10]`	Create array named `cart` with 10 elements indexed 0-9
`cart[i] = 50`	Set an array element using a variable as the index.

A `WHILE`-loop can be used to access every element in a one-dimensional array:

```
index = 0
ARRAY grades [10]
WHILE (index < 10)
     grades[i] = 90
     index = index + 1
ENDWHILE
```

19. What set of code correctly initializes all elements of the array `ara` to the value 0? Assume the declaration `ARRAY ara[5]`

A)
```
ndx = 0
WHILE (ndx < 5)
   ara[ndx] = 0
   ndx = ndx + 1
ENDWHILE
```

B)
```
ndx = 1
WHILE (ndx < 5)
   ara[ndx] = 0
   ndx = ndx + 1
ENDWHILE
```

C)
```
ndx = 0
WHILE (ndx <= 5)
   ara[ndx] = 0
   ndx = ndx + 1
ENDWHILE
```

D)
```
ndx = 1
WHILE (ndx <= 5)
   ara[ndx] = 0
   ndx = ndx + 1
ENDWHILE
```

A two-dimensional array hold a group of related items that are accessed by two indices. One example two-dimensional array would be the grades for an entire class of students.

`ARRAY grades [25][10]`	Array named `grades` for 25 students, with 10 grades each.
`grades[5][0] = 100`	The *sixth* student's *first* grade is 100. (Remember array indices start at 0.)

Nested `WHILE`-loops can be used to access every element in a two-dimensional array:

```
studentIndex = 0
gradeIndex = 0
ARRAY grades [10][5]
WHILE (studentIndex < 10)
     WHILE (gradeIndex < 5)
         grades[studentIndex][gradeIndex] = 90
         gradeIndex = gradeIndex + 1
     ENDWHILE
     studentIndex = studentIndex + 1
ENDWHILE
```

20. A two-dimensional array's data is usually loaded using:

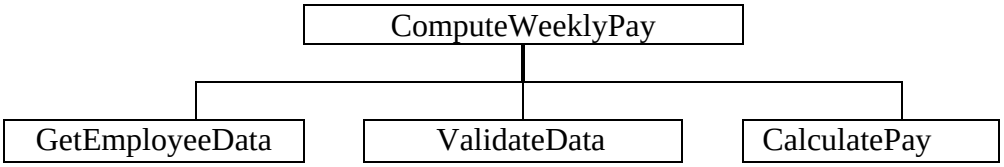
    A) a single loop which computes both subscripts
    B) a single loop with a single subscript
    C) two nested loops
    D) no loops

G. Modularization

Recognize and understand the concept of a hierarchy chart that demonstrates the concepts of modularization, and scalability of software.

Complex programming projects need to be broken down into smaller subtasks, called *modules* (or functions, methods, or subroutines). Each module performs a single, specific task. A module has a single entry point and a single exit point. The name of the module is a verb phrase that describes its function, such as `enterCustomerData` or `printInvoice`.

A *hierarchy chart* graphically shows the relationship of the modules to one another. For example, consider the program `ComputeWeeklyPay` that calculates an employee's pay for the week. The tasks are 1) to get the input (employee name, rate, hours worked), 2) validate the input, and 3) calculate the pay. The hierarchy chart would be:



Each programming language has its own way to indictate that a module should be executed. In this study guide, we will use **`GOSUB` *`moduleName`*** to indicate that you should execute the module (subroutine), then resume execution after the call to the module.

21. Pick the choice that shows what would be printed by the following: (Assume variables `count` and `sum` are accessible in both `Main` and the module `Count2`)

```
Main
ct = 0
sum = 0
WHILE (ct < 3)
    GOSUB Count2
    ct = ct + 1
ENDWHILE
PRINT ct, sum
```

```
module Count2
    newCt = 0
    WHILE (newCt < 4)
        sum = sum + 4
        newCt = newCt + 2
    ENDWHILE
    RETURN
```

A) 3 24 B) 3 36 C) 4 24 D) 4 36

H. Common algorithms

Understand and identify common algorithms for searching and sorting. Be able to walk through algorithms that implement searching and sorting.

For these examples, our "database" is an *array* with our information, such as customer records.

Sorting

Most organizations need to present their data in a particular order, such as alphabetical or numerical. The process of putting data into order is called *sorting*. Common algorithms for sorting include *bubble sort* and *selection sort.*

Bubble Sort

This algorithm is also called *sinking sort*. Items bubble up to the top or sink to the bottom.

This algorithm makes multiple passes through the array. Adjacent items are compared, and they are swapped if they are out of order. At the end of the first pass, the largest element has sunk down to the bottom to the array. After the second pass, the second largest element is in the next-to-last position in the array, etc.

```
ARRAY data[100]
pass = 100
WHILE (pass >= 2)
     j = 1
     WHILE (j <= i-1)
          IF (data[j] > data[j+1]) THEN
               temp = data[j]
               data[j] = data[j+1]
               data[j+1] = temp
          ENDIF
          j = j + 1
     ENDWHILE
     pass = pass - 1
ENDWHILE
```

Selection Sort

This algorithm scans the array until an out-of-order element is found. The array is then searched backwards from that position to find the correct location for the out-of-order element. Elements that are in their correct relative order are moved down one location in the array to make room for the element being put in its sorted location.

```
ARRAY data[100]
numElements = 100
i = 1
WHILE (i <= numElements - 1)
     IF (data[i] > data[i+1]) THEN
          temp = data[i+1]
          j = 1
          WHILE (j >= 1 AND data[j] > temp)
               data[j+1] = data[j]
               j = j - 1
          ENDWHILE
          data[j+1] = temp
     ENDIF
     i = i + 1
ENDWHILE
```

Searching

Common algorithms for looking for an item in your database are the *linear search* and the *binary search*. The "search space" is the set of items in which you're looking for your item. Searches return a particular value, usually –1, if the item is not found.

Linear Search

Start at the beginning of your array, and continue searching until you find your item.

Pros: Easy to write.
Cons: You spend time looking at each and every item; can't eliminate any items automatically from your search space.

```
ARRAY myArray[100]
index = 1
numElements = 100
WHILE (index <= numElements)
     IF (key = myArray[index] THEN
          return index
     ENDIF
     index = index + 1
ENDWHILE
RETURN -1
```

Binary Search

The array that you're searching must be sorted first.

Start at the middle of your search space, and compare what you're looking for to that item. If your item (the key) is less than that middle item, continue your search in the lower half of the array. If the key is greater than the middle item, you'll continue your search in the upper half of the array. Compare the key to the middle item of the remaining search space, and eliminate the upper or lower half of the remaining space. Continue seraching the middle item of the sub-array until either you find the item, or there are no potential candidates left.

Pros: With each comparison, you eliminate half of the remaining items from your search space.
Cons: Array must be sorted for this algorithm to work.

```
ARRAY myArray[100]
lowIndex = 1
highIndex = 100
WHILE (lowIndex <= highIndex)
     middleIndex = (lowIndex + highIndex) / 2
     IF (key = myArray[middleIndex]) THEN
          RETURN middleIndex
     ELSE
          IF (key  >  myArray[middleIndex]) THEN
                lowIndex = middleIndex + 1
          ELSE
                highIndex = middleIndex – 1
          ENDIF
     ENDIF
ENDWHILE
RETURN –1
```

22. Which answer best describes the following code?

```
ARRAY myArray[100]
index = 1
numElements = 100
WHILE (index <= numElements)
     IF (key = myArray[index] THEN
          return index
     ENDIF
     index = index + 1
ENDWHILE
RETURN -1
```

A) linear search of elements in an array
B) binary search of elements in an array
C) bubble sort of elements in an array
D) insertion sort of elements in an array

I. Software development lifecycle

Understand the software development lifecycle that includes requirements, design, code, test, and maintain. Understand the necessity to have quality assurance in software development.

The software development cycle begins with the specification of the requirements. The "specs" nail down the business functionality of the software – what it does, and how it meets the customers' needs. This phase is followed by the design – *how* we'll satisfy the requirements. Well thought-out requirements and design lead to straight-forward development of the product, because many potential obstacles have already been discovered and solved. Haphazard design leads to buggy software and late nights trying to work around problems that would have been easier to fix in the requirements or design stage.

Concurrent with specification, design, and development is the documentation. Two different sets are needed: one for internal use, and one for the customer. The requirements, design, and implementation details are documented for both quality assurance and maintenance. How to install and run the software, and what to do when there's a problem, is documented for the customer.

QA is a separate department that verifies that the product delivered works and meets the customer's needs. Programmers know too well how the software works, and are likely to subconsciously avoid problems. Independent QA engineers, on the other hand, are more likely to find unexpected bugs or usability problems in the software.

Software development is a cycle – as soon as the product is shrink-wrapped, the maintenance begins ... with requirements. The requirements may be merely to fix reported bugs, but more likely defines the next release of the software.

23. Independent quality assurance engineers are used to test if software meets the specifications. Why should the quality assurance personnel be different from the programmers?

A) Programmers cannot be trusted to report their own errors.
B) QA engineers are more likely to find unexpected bugs or usability problems in the software.
C) Programmers are too busy and too highly paid to do trivial tasks.
D) QA professionals do not know how to write software.

J. Problem solving using formal logic. (Algorithm development)
Create a solution to a problem described in English. Complexity of this problem statement will include the requirement to nest at least one repetition construct with selection constructs. For example, a correct solution will require the placement of an IF – THEN construct within a WHILE construct.

In general, we use selection constructs to test the validity of the data. We use repetition constructs to perform the same set of actions for a number of different data sets.

24. There is a data file containing records. Each record has an employee number, the hours they worked during a week, and their hourly wage. The hourly wage value must be in the range of 5 – 60 in order for it to be valid. You are asked to write a program that will produce a weekly salary report. Invalid data should not be included on your report. Which of the following constructs will appear in your program?

    A) There will be a selection construct inside of a selection construct
    B) There will be a selection construct inside of a repetition construct
    C) There will be a repetition construct inside of a selection construct
    D) There will be a repetition construct inside of a repetition construct

25. You are writing an interactive program to calculate the overall gas mileage for several tanks of gas. The user of the program will provide values that represent the gallons of gas used, and the corresponding miles traveled. When the user has entered all of the data, the user will then enter the value –1.

    Which of the following pseudocode solutions best solves the problem?

A)
```
Prompt for a gallons value
WHILE (gallons NE –1)
      Prompt for a miles value
      Accumulate the gallons, and miles
      Prompt for a gallons value
ENDWHILE
Calculate the average miles per gallon
Output the average miles per gallon
```

B)
```
WHILE (gallons NE –1)
      Prompt for a gallons value
      Prompt for a miles value
      Calculate the miles per gallon
ENDWHILE
Calculate the average miles per gallon
Output the average miles per gallon
```

C)
```
Prompt for a gallons value
WHILE (gallons NE –1)
      Prompt for a gallons value
      Calculate the miles per gallon
      Accumulate the miles per gallon
      Prompt for a gallons value
ENDWHILE
Calculate the average miles per gallon
Output the average miles per gallon
```

D)
```
Prompt for a gallons value
Prompt for a miles value
WHILE (gallons NE –1)
      Calculate the miles per gallon
      Accumulate the miles per gallon
ENDWHILE
Calculate the average miles per gallon
Output the average miles per gallon
```

26. You are writing an interactive program to tally orders for boxes of cookies. The user will enter 1 for Thin Mints or 2 for Shortbread. Then the user will enter the number of boxes ordered. Keep track of how many boxes of each cookie the user has ordered. When the user has entered all of the orders, the user will then enter the value –1.

    Which of the following pseudocode solutions best solves the problem?

A)
```
Prompt for box type
WHILE (box type NE –1)
      IF (box type = 1) THEN
            Accumulate thin mint boxes
      ELSE
            Accumulate shortbread boxes
      ENDIF
      Prompt for number of boxes
ENDWHILE
Output the number of thin mint and shortbread boxes
```

B)
```
Prompt for box type
Prompt for number of boxes
WHILE (box type NE –1)
      IF (box type = 1) THEN
            Accumulate thin mint boxes
      ELSE
            Accumulate shortbread boxes
      ENDIF
ENDWHILE
Output the number of thin mint and shortbread boxes
```

C)
```
Prompt for box type
WHILE (box type NE –1)
      Prompt for number of boxes
      IF (box type = 1) THEN
            Accumulate thin mint boxes
      ELSE
            Accumulate shortbread boxes
      ENDIF
      Prompt for box type
ENDWHILE
Output the number of thin mint and shortbread boxes
```

D)
```
WHILE (box type NE –1)
      Prompt for box type
      Prompt for number of boxes
      IF (box type = 1) THEN
            Accumulate thin mint boxes
      ELSE
            Accumulate shortbread boxes
      ENDIF
ENDWHILE
Output the number of thin mint and shortbread boxes
```

Solutions to Sample Questions

1. What is the result of converting the decimal number 37 to binary? **B**

   binary places:

128	64	32	16	8	4	2	1
0	**0**	**1**	**0**	**0**	**1**	**0**	**1**

   (0*128) + (0*64) + (1 * 32) + (0*16) + (0*8)+ (1 * 4) + (0*2) + (1 * 1) = 37

   A) `1010 0101` **B) `0010 0101`** C) `0111 0101` D) `0101 0101`

2. What is the result of converting the decimal number 92 to binary? **D**

   A) `0101 1000` B) `0101 0101` C) `0011 1100` **D) `0101 1100`**

   binary places:

128	64	32	16	8	4	2	1	
0	**1**	**0**	**1**	**1**	**1**	**0**	**0**	64 + 16 + 8 + 4 = 92

3. What is the result of converting the hex number 0x37 to decimal? **C**

   A) `37` B) `1011 0100` **C) `55`** D) `775`

   Hex places: 16 1
   3 **7**
   (3 * 16) + (7 * 1) = 55

4. What is the result of converting the hex number 0x2A to decimal? **A**

   A) `42` B) `0010 1010` C) `25` D) `775`

   Hex places: 16 1
   2 **A**

   (2 * 16) + (A * 1) = 32 + 10 = 42

5. A file containing a few pages of text is stored on a hard disk. The file is most likely to require what amount of storage space? **B**

   A) A few Gigabytes **B) A few Kilobytes** C) A few Megabytes D) A few bytes

   A few pages of text would have several thousand characters, and so would contain a few kilobytes.

6. A desktop PC is for sale in a local computer shop. The sign indicates that it has 700 MHz. This refers to: **C**

   A) sound card quality
   B) 700 megabytes of Random Access Memory
   C) the processor speed
   D) a hard disk containing 700 megabytes of storage

7. What is the value of **num** after the following statement has been executed? **A**

```
num = 30 + 3 * 2        multiplication has higher precedence than addition
num = 30 + 6
```

A) 36 B) 66 C) 92 D) 180

8. What is the value of **num** after the following statement has been executed? **D**

```
num = 30 % 7        take 4 groups of 7 (4*7=28) out of 30, with 2 left over
```

A) 0 B) 210 C) 4 **D) 2**

9. What is the value of **num** after the following statements have been executed? **D**

```
x = 5
y = x + 2              y = 5 + 2, so y = 7
x =  x + 1             x = 5 + 1, so x = 6
num = x + y            num = 7 + 6
```

A) 8 B) 12 C) 15 **D) 13**

10. Given that `b = 3, c = 6,` what is the value of the expression? **B**

```
b > 5 AND c > 5
3 > 5     6 > 5
false     true
false AND true is false
```

A) true **B) false** C) it depends D) can't know

11. Given that `c = 5, d = 9,` what is the value of the expression? **A**

```
c < 4 OR d > 6
5 < 4    9 > 6
false    true
false OR true is true
```

A) true B) false C) it depends D) can't know

12. What is the value of **num** after the following statements have been executed? **B**

```
x = 5
y = x + 1                  y = 5 + 1 so, y = 6
IF (x < y) THEN            5 < 6 is true
      x = x + 1            x = 5 + 1 so, x = 6
ENDIF
num = x + y                num = 6 + 6
```

A) 11 **B) 12** C) 13 D) 15

13. What will be displayed by the following statements? **D**

```
x = 3
y = x * -1                          y = -3
y = y * y                           y = 9
IF (x < y) AND (y < 0) THEN         3 < 9 AND y < 0
                                    true  AND false is false
      PRINT x                       this statement is not executed
ENDIF
```

A) -6 B) 3 C) 36 **D) nothing**

14. What is the value of **num** after the following statements have been executed? **C**

```
x = 2
y = x * x + 1                       y = 2*2 + 1, so, y = 5
IF (y <= 5) THEN                    true
      num = x + y                   num = 2 + 5
ELSE
      num = x – y
ENDIF
```

A) -4 B) -3 **C) 7** D) 8

15. What will be displayed by the following statements?

```
x = 5
y = x * -1 + 2                      y = -5 + 2 = -3
IF (x < y) AND (y > 0) THEN         5 < -2 is false
      PRINT x * x, y * y
ELSE
      PRINT y * y, x * x            PRINT -3*-3, 5*5
ENDIF
```

A) 9 25 B) 25 9 C) 25 25 D) 25 –25

16. What is the value of **num** after the following statements have been executed? **B**

```
x = 1
y = 8
z = 5
IF (x < 7)THEN                      1 < 7 is true
      IF (y < 5) THEN               8 < 5 is false, execute ELSE statement
            num = 1
      ELSE
            num = 2                 num = 2
      ENDIF
ELSE
      IF (z > 5) THEN
            num = 3
      ELSE
            num = 4
      ENDIF
ENDIF
```

A) 1 **B) 2** C) 3 D) 4

17. What will be displayed by the following statements? **A**

```
x = 1
IF (x < 7 AND x*3 > 2) THEN          1 < 7 AND 3 > 2
                                     true  AND true is true
      num = 1                        execute num = 1
ELSE
      num = 3
ENDIF

CASE num
   1:       PRINT 4                  ◄ this statement is selected
   2:       PRINT 3
   3:       PRINT 2
   4:       PRINT 1
ENDCASE
```

A) 4 B) 3 C) 2 D) 1

18. How many times will the `PRINT` statement be executed in the following code? **B**

```
outer = 1
WHILE (outer < 2)
      inner = 1
      WHILE (inner <= 2)
            PRINT inner * outer
            inner = inner + 1
      ENDWHILE
      outer = outer + 1
ENDWHILE
```

Execution trace:

outer	(outer<2)?	inner	(inner<=2)?	output
1	true			
		1	true	PRINT 1*1
		2	true	PRINT 2*1
		3	false	
2	false			

A) 0 **B) 2** C) 4 D) 6

19. What set of code correctly initializes all elements of the array `ara` to the value 0? Assume the declaration `ARRAY ara[5]`. **A**

A)
```
ndx = 0
WHILE (ndx < 5)
   ara[ndx] = 0
   ndx = ndx + 1
ENDWHILE
```

B)
```
ndx = 1
WHILE (ndx < 5)
   ara[ndx] = 0
   ndx = ndx + 1
ENDWHILE
```

C)
```
ndx = 0
WHILE (ndx <= 5)
   ara[ndx] = 0
   ndx = ndx + 1
ENDWHILE
```

D)
```
ndx = 1
WHILE (ndx <= 5)
   ara[ndx] = 0
   ndx = ndx + 1
ENDWHILE
```

20. A two-dimensional array's data is usually loaded using: **C**

    A. a single loop which computes both subscripts
    B. a single loop with a single subscript
    C. two nested loops
    D. no loops

21. Pick the choice that shows what would be printed by the following. Assume variable `sum` is accessible in both `Main` and the module `Count2`.

```
Main
ct = 0
sum = 0
WHILE (ct < 3)
     GOSUB Count2
     ct = ct + 1
ENDWHILE
PRINT ct, sum
```

```
module Count2
     newCt = 0
     WHILE (newCt < 4)
          sum = sum + 4
          newCt = newCt + 2
     ENDWHILE
     RETURN
```

A) **3 24** B) 3 36 C) 4 24 D) 4 36

Execution trace:

ct	(ct<3)?	newCt	(newCt<4)?	sum
0				0
	true			
		0	true	0+4=4
		2	true	4+4=8
		2+2=4	false	
1	true			
		0	true	8+4=12
		2	true	12+4=16
		2+2=4	false	
2	true			
		0	true	16+4=20
		2	true	20+4=**24**
		2+2=4	false	
3	false			

22. Which answer best describes the following code? **A)**

```
ARRAY myArray[100]
index = 1
numElements = 100
WHILE (index <= numElements)
     IF (key = myArray[index] THEN
          return index
     ENDIF
     index = index + 1
ENDWHILE
RETURN -1
```

A) Linear search of elements in an array
B) Binary search of elements in an array
C) Bubble sort of elements in an array
D) Selection sort of elements in an array

23. Independent quality assurance engineers are used to test if software meets the specifications. Why should the quality assurance personnel be different from the programmers? **B)**

    A. Programmers cannot be trusted to report their own errors.
    B. QA engineers are more likely to find unexpected bugs or usability problems in the software.
    C. Programmers are too busy and too highly paid to do trivial tasks.
    D. QA professionals do not know how to write software.

24. There is a data file containing records. Each record has an employee number, the hours they worked during a week, and their hourly wage. The hourly wage value must be in the range of 5 – 60 in order for it to be valid. You are asked to write a program that will produce a weekly salary report. Invalid data should not be included on your report. Which of the following constructs will appear in your program?

    A) There will be a selection construct inside of a selection construct
    B) There will be a selection construct inside of a repetition construct
    C) There will be a repetition construct inside of a selection construct
    D) There will be a repetition construct inside of a repetition construct

25. You are required to create an interactive program to calculate the overall gas mileage for several tanks of gas. The user of the program will provide values that represent the gallons of gas used, and the corresponding miles traveled. When the user has entered all of the data, the user will then enter the value –1.

    Which of the following pseudocode solutions best solves the problem? **A)**

A)
```
Prompt for a gallons value
WHILE (gallons NE –1)
      Prompt for a miles value
      Accumulate the gallons, and miles
      Prompt for a gallons value
ENDWHILE
Calculate the average miles per gallon
Output the average miles per gallon
```

26. You are writing an interactive program to tally orders for boxes of cookies. The user will enter 1 for Thin Mints or 2 for Shortbread. Then the user will enter the number of boxes ordered. Keep track of how many boxes of each cookie the user has ordered. When the user has entered all of the orders, the user will then enter the value –1.

    Which of the following pseudocode solutions best solves the problem? **C)**

C)
```
Prompt for box type
WHILE (box type NE –1)
      Prompt for number of boxes
      IF (box type = 1) THEN
            Accumulate thin mint boxes
      ELSE
            Accumulate shortbread boxes
      ENDIF
      Prompt for box type
ENDWHILE
Output the number of thin mint and shortbread boxes
```