**Multiple Choice**

1. The first step in creating a Physical Data Flow Diagram is what?
2. Update the metadata in the CASE repository
3. Draw a human-machine boundary
4. Add implementation references
5. Add system-related data stores, data flows and processes
6. Update the data elements in the data flows

Ans: c

Reference: The Physical Data Flow Diagram

Difficulty: medium

1. The second step in creating a Physical Data Flow Diagram is what?
2. Update the metadata in the CASE repository
3. Draw a human-machine boundary
4. Add implementation references
5. Add system-related data stores, data flows and processes
6. Update the data elements in the data flows

Ans: b

Reference: The Physical Data Flow Diagram

Difficulty: medium

1. The third step in creating a Physical Data Flow Diagram is what?
2. Update the metadata in the CASE repository
3. Draw a human-machine boundary
4. Add implementation references
5. Add system-related data stores, data flows, and processes
6. Update the data elements in the data flows

Ans: d

Reference: The Physical Data Flow Diagram

Difficulty: medium

1. The fourth step in creating a Physical Data Flow Diagram is what?
2. Update the metadata in the CASE repository
3. Draw a human-machine boundary
4. Add implementation references
5. Add system-related data stores, data flows and processes
6. Update the data elements in the data flows

Ans: e

Reference: The Physical Data Flow Diagram

Difficulty: medium

1. The fifth step in creating a Physical Data Flow Diagram is what?
2. Update the metadata in the CASE repository
3. Draw a human-machine boundary
4. Add implementation references
5. Add system-related data stores, data flows and processes
6. Update the data elements in the data flows

Ans: a

Reference: The Physical Data Flow Diagram

Difficulty: medium

1. The Human-Machine Boundary is what?
2. Is the keyboard and screen
3. Builds on research in ergonometric
4. Is a line drawn on the physical DFD to separate human action from automated processes
5. Is a part of developing the HCI interface
6. Separates where manual processes are separated by human completed processes

Ans: c

Reference: The Physical Data Flow Diagram

Difficulty: medium

1. In most automated cases, data stores from logical DFDs will be converted to what?
2. Binary tables
3. DVD disks
4. A thumb drive
5. Into database files / tables
6. Encrypted hexadecimal values

Ans: d

Reference: The Physical Data Flow Diagram

Difficulty: medium

1. By definition, external entities on the DFD are what?
2. Are used as the starting point for the physical data flow diagram
3. Are outside the scope of the system
4. Will be the top of the structure chart
5. Will become database table entries
6. Are updated with metadata to become part of the physical DFD

Ans: b

Reference: The Physical Data Flow Diagram

Difficulty: medium

1. Every part of a system that is not automated will what?
2. Be ignored in the implementation of the system
3. Be drawn as only logical DFDs not physical DFDs
4. Be outside the human-machine boundary
5. Go back through the analysis phase to become automated
6. Will be drawn as diamond shapes on the structure charts

Ans: c

Reference: the Physical Data Flow Diagram

Difficulty: medium

1. In determining if a process is to be automated, the project team will do all of the following EXCEPT:
2. Weigh the costs
3. Redraw the logical DFD as a combined logical/physical DFD
4. Determine benefits
5. Evaluate the efficiency
6. Consider the integrity of the process to the system

Ans: b

Reference: The Physical Data Flow Diagram

Difficulty: medium

1. Beth is considering a simple process for immediate orders. When processing an immediate order, a phone clerk writes the order on a paper form. Which of the following might be a reason NOT to automate this process?
2. The clerk could make errors when writing the information wrong on the paper form
3. The paper form could be lost when sent to the order filling /shipping area
4. The writing on the paper form could be hard to decipher
5. The costs to automate might be significantly higher than doing it manually
6. The time to get the paper form from the clerk’s desk to the order filling / shipping area is one day or longer

Ans: d

Reference: The Physical Data Flow Diagram

Difficulty: medium

1. When changing a logical DFD into a physical DFD, it might be necessary to do what?
2. Add system-related data stores, data flows and processes
3. Normalize the logical DFD into 3NF
4. Create the user-interface with smaller fonts
5. Change the system architecture to three-tiered architecture
6. Delete extraneous data stores and delete extraneous data flows

Ans: a

Reference: The Physical Data Flow Diagram

Difficulty: medium

1. When you are updating the data elements in the data flows (when creating a physical DFD) you might need to do what?
2. Return to users to interview them about the physical data flows
3. Update the original cost/benefit analysis to reflect the physical storage
4. Do a technology analysis
5. Add physical data elements to the metadata descriptions in the CASE repository
6. Do formal benchmarking of the data flows

Ans: d

Reference: The Physical Data Flow Diagram

Difficulty: medium

1. What is pseudocode?
2. The same as structured English
3. A technique similar to structured English
4. A subset of the Java programming language
5. A coding environment sponsored by Oracle
6. The term for designing language prototyping screens with Visual Basic or HTML

Ans: b

Reference: Designing Programs

Difficulty: medium

1. A general suggestion about using couples in drawing structure charts is what?
2. The use of many couples clarify the processing
3. It is best to be conservative when applying couples to your diagram
4. To use ‘combination’ couples when both data couples and control couples are needed
5. You should have at least twice as many afferent couples as efferent couples
6. Use only data couples and no control couples

Ans: b

Reference: Structure Chart

Difficulty: medium

1. The three types of basic processes on a process model are which of the following?
2. Sequence, selection and iteration processes
3. Navigation, status and work processes
4. Afferent, central and efferent processes
5. Batch, online and real time processes
6. Singular, bilateral and library processes

Ans: c

Reference: Building the Structure Chart

Difficulty: hard

1. Generally, transaction structures on a structure chart will have what?
2. Many afferent processes
3. Many efferent processes
4. Many data couples
5. Many control couples
6. Many conditional couples

Ans: b

Reference: Building the Structure Chart

Difficulty: hard

1. Generally, transform structures on a structure chart will have what?
2. Many afferent processes
3. Many efferent processes
4. Many data couples
5. Many control couples
6. Many conditional couples

Ans: a

Reference: Building the Structure Chart

Difficulty: hard

1. As a structure chart is constructed, it is generally best to build modules with which of the following?
2. High cohesion
3. Highly coupled
4. High fan-out
5. Have the word “and” in the module title
6. A high level of coincidental cohesion

Ans: a

Reference: Design Guidelines

Difficulty: medium

1. What is pseudocode?
2. A language popularized by Feinstein and Longenecker in the 1990’s
3. A detailed outline of the lines of code that need to be written
4. A ‘pretend’ syntax inherent in all CASE tools
5. The translation of code into ASCII
6. A dialect of Visual Basic only used in program design

Ans: b

Reference: Program Specification

Difficulty: medium

1. Reusable modules, which are represented in the structure chart as rectangles with vertical lines on both sides, may often appear several times in a structure chart. These are called \_\_\_\_\_ modules.
2. conditional
3. control
4. library
5. loop
6. off-page connector

Ans: c

Reference: Structure Chart

Difficulty: medium

1. There are two symbols that describe special types of control in a structure chart. They are a curved arrow and a diamond. These symbols represent \_\_\_\_\_ and \_\_\_\_\_.
2. connector, conditional line
3. control, subordinate
4. library module, conditional line
5. loop, conditional line
6. loop, connector

Ans: d

Reference: Structure Chart

Difficulty: medium

1. In a structure chart, the element that communicates that a message or a system flag is being passed from one module to another is known as a(n) \_\_\_\_\_.
2. conditional line
3. connector
4. control couple
5. data couple
6. loop

Ans: c

Reference: Structure Chart

Difficulty: medium

1. In a structure chart, the purpose of a control couple is to do what?
2. Pass parameters from a subordinate module to the control module
3. Pass parameters from the control modules to a subordinate module
4. Pass data from a subordinate module to the control module
5. Pass data from the control module to a subordinate module
6. Chaperone the dance for programmers on the project

Ans: a

Reference: Structure Chart

Difficulty: medium

1. \_\_\_\_\_ refers to how well the lines of code within each module in a structure chart relate to each other.
2. Calculation
3. Cohesion
4. Control
5. Coupling
6. Fan-in

Ans: b

Reference: Design Guidelines

Difficulty: medium

1. In a structure chart, the element that is drawn as an empty circle with an attached arrow is known as a(n) \_\_\_\_\_.
2. conditional line
3. connector
4. control couple
5. data couple
6. module

Ans: d

Reference: Structure Chart

Difficulty: medium

1. In a structure chart, a *transaction structure* \_\_\_\_\_.
2. contains a control module that calls several subordinate modules in sequence, after which something “happens”
3. contains a control module that calls subordinate modules, each handling a particular transaction
4. contains a transactional loop
5. is a subordinate module that handles a particular transaction
6. is subordinate to subordinal modules

Ans: b

Reference: Design Guidelines

Difficulty: medium

## True / False

1. When designing programs, structure charts may have on-page and off-page connectors.

Ans: True

Reference: Structure Chart

Difficulty: medium

1. When designing programs, structure charts help analysts create programs that are easy to understand and maintain.

Ans: True

Reference: Structure Chart

Difficulty: medium

1. When designing programs, structure charts are arranged in a hierarchical format that implies sequence.

Ans: True

Reference: Structure Chart

Difficulty: medium

1. When designing programs, structure charts show all the components of code that must be included in a program at a high level.

Ans: True

Reference: Structure Chart

Difficulty: medium

1. When designing programs, structure charts are generally implemented with control flags that pass from the control modules to the subordinate modules.

Ans: False

Reference: Structure Chart

Difficulty: medium

1. Structure charts show the user interface.

Ans: False

Reference: Structure Chart

Difficulty: medium

1. Structure charts emphasize structure and reusability.

Ans: True

Reference: Structure Chart

Difficulty: medium

1. Structure charts show selection.

Ans: True

Reference: Structure Chart

Difficulty: medium

1. Structure charts show couples.

Ans: True

Reference: Structure Chart

Difficulty: medium

1. Structure charts show sequence.

Ans: True

Reference: Structure Chart

Difficulty: medium

1. Physical DFDs show additional details, such as what tables in the database replace data stores on the logical DFD

Ans: True

Reference: Designing Programs

Difficulty: medium

1. Programmers can take the logical DFDs and directly implement into code by compiling the metadata in the CASE tools.

Ans: False

Reference: Designing Programs

Difficulty: medium

1. Analysts can build in the three structures of sequence, selection and iteration into structure charts.

Ans: True

Reference: Designing Programs

Difficulty: medium

1. Analysts can use structure charts to design programming logic.

Ans: True

Reference: Designing Programs

Difficulty: medium

1. Analysts design programs in the design phase of the SDLC, programmers code programs in the Implementation phase.

Ans: True

Reference: Designing Programs

Difficulty: medium

1. Because project teams rely more on packaged software, program design is no longer needed.

Ans: False

Reference: Introduction

Difficulty: easy

1. Because (a) preexisting code needs to be understood, organized, and pieced together; and (b) it is still common for the project team to have to write some code and adapt packages to the business environment, it is therefore good for analysts to fully understand program design.

Ans: True

Reference: Introduction

Difficulty: Medium

1. The Human-Machine boundary is part of the Human-Computer Interface into usability systems.

Ans: False

Reference: The Physical Data Flow Diagram

Difficulty: medium

1. Micah is adding implementation resources to change a logical DFD into a physical DFD. He should describe databases, files, tables, and processes as they will be implemented on the computer.

Ans: True

Reference: The Physical Data Flow Diagram

Difficulty: medium

1. There are some ‘system-related’ data stores, data flows and processes that must be added when creating the physical DFD.

Ans: True

Reference: The Physical Data Flow Diagram

Difficulty: easy

1. Many of the data stores in the logical DFD will be changed into encrypted word processing documents when a physical DFD is developed.

Ans: False

Reference: The Physical Data Flow Diagram

Difficulty: medium

1. When drawing the human-machine boundary, all processes in the physical DFD will be automated, so only external entities will be excluded.

Ans: False

Reference: The Physical Data Flow Diagram

Difficulty: medium

1. An audit control / audit logfile might be a system related process, data flow and data store that would be added when creating physical DFDs.

Ans: True

Reference: The Physical Data Flow Diagram

Difficulty: medium

1. When creating the physical DFD, you may want to capture system information like date and time of update and user-id of the person who did the update.

Ans: True

Reference: The Physical Data Flow Diagram

Difficulty: easy

1. A temptation when moving from analysis to design of a system is to jump right into coding.

Ans: True

Reference: Designing Programs

Difficulty: medium

1. A structure chart shows sequence, selection and data flows.

Ans: False

Reference: Structure Chart

Difficulty: Medium

1. A structure chart is composed of modules that work together to form a program.

Ans: True

Reference: Structure Chart

Difficulty: medium

1. A control module can also be called a library module and is a set of reusable code.

Ans: False

Reference: Structure Chart

Difficulty: medium

1. A diamond on a structure chart shows sequence – like a baseball player would go to first base, then second, third and home.

Ans: False

Reference: Structure Chart

Difficulty: medium

1. Couples in structure charts always show two modules that are executed together and are drawn with horizontal lines at the top and bottom.

Ans: False

Reference: Structure Chart

Difficulty: medium

1. Data couples are shown by arrows with empty circles and show how data flows between modules.

Ans: True

Reference: Structure Chart

Difficulty: medium

1. Control couples show the passing of parameters or system related messages between modules (like ‘end-of-file’).

Ans: True

Reference: Structure Chart

Difficulty: medium

1. A structure chart is often completed in one attempt.

Ans: False

Reference: Applying the Concepts at DrōnTeq Difficulty: medium

1. The three basic kinds of processes on a process module are afferent, central, and efferent.

Ans: True

Reference: Building the Structure Chart

Difficulty: medium

1. An afferent process is an input process on a structure chart.

Ans: True

Reference: Building the Structure Chart

Difficulty: medium

1. An effective process is an output process on a structure chart.

Ans: False

Reference: Building the Structure Chart Difficulty: medium

1. A transaction structure in a structure chart contains a control module that calls subordinate modules and frequently occur with menus.

Ans: True

Reference: Building the Structure Chart

Difficulty: medium

1. Factoring is the process of separating out a function from a module in order to create a module of its own.

Ans: True

Reference: Design Guidelines

Difficulty: medium

1. A signal of poor cohesion could be the presence of control flags that are passed down to subordinate modules.

Ans: True

Reference: Design Guidelines

Difficulty: medium

1. Functional, coincidental, and temporal are all types of cohesion.

Ans: True

Reference: Design Guidelines

Difficulty: medium

1. Fan-In describes the number of subordinates that communicate with the control module.

Ans: False

Reference: Design Guidelines

Difficulty: medium

1. Generally, transaction structures occur at lower levels of a structure chart.

Ans: False

Reference: Building the Structure Chart

Difficulty: medium

1. A good indication of needing a transaction structure on a structure chart occurs when a DFD shows a single data flow entering a process that produces multiple data flows as outputs.

Ans: True

Reference: Building the Structure Chart

Difficulty: medium

1. Jim is working from a leveled DFD and creating structure charts. He is finding that the lower levels of the DFD generally correspond to transform structures.

Ans: True

Reference: Building the Structure Chart

Difficulty: medium

## Essays

1. Compare and contrast cohesion and coupling.

Answer

Cohesion is how well a module does processing – a module should do only one task; coupling is how closely modules are interrelated – the less the better. So, modules should be highly cohesive (only one task) and loosely coupled (little or no interaction).

Reference: Design Guidelines

Difficulty: medium

1. It seems a lot of work to create structure charts, with high cohesion, loosely coupled, with high fan-in, low-fan-out, with transaction structures and transform structures. Why should an analyst bother developing structure charts and instead just pass on the specifications to the programmer?

Answer

Analysts do the assigning of application development to programmers. You want programs that are easily maintained, modular, flexible, and satisfy the users. That is why structure charts and subsequent program design actions are so important.

Reference: Entire chapter

Difficulty: medium

1. What is the difference between transaction structures and transform structures on a structure chart?

Answer

Transaction structures generally are found in the higher levels of a DFD and are where a module performs one of a group of individual transactions and generally have few inputs and a lot of outputs; transform structures have control modules that call several subordinate modules in sequence.

Reference: Building the Structure Chart

Difficulty: medium

1. What is a ‘top-down approach’ and why is it important when designing programs?

Answer

Starting with the big picture, refining it to more detail and to a fully described scenario (the example in the text was to drive to a specific location; start with the general directions (Virginia); then the regional directions (south-central Virginia); then local directions (315 N. Elm Street, Charlottesburg, VA). Top-down approaches help the analyst create modules that will get translated into flexible, maintainable, well-designed code.

Reference: Designing Programs

Difficulty: medium

1. Structure charts show: sequence, selection and iteration. What are these factors?

Answer

* Sequence shows in what order components are invoked
* Selection is under what condition a module is invoked Iteration is how oft en a component is repeated

Reference: Structure Chart

Difficulty: medium

1. What is Fan-in and Fan-out?

Answer

Fan-in describes the number of control modules that communicate with a subordinate; a module with high-fan-in has many different control modules that call it (it may be a library module) – and suggests well-written generic code; fan-out is the concept of ‘span of control’. If too many subordinate modules are associated with a single control, it might be hard to maintain and manage.

Reference: Design Guidelines

Difficulty: medium

1. Summarize the quality checks that should be performed when reviewing structure charts.

Answer

Design the system so that there is a high fan-in structure. Create library modules whenever possible. Limit the span of control of the control modules to no more than seven subordinates. Make sure that each module performs only one function. The modules should sparingly share information, and make sure that the data couples that are passed are actually used by the accepting module. Control couples should be passed from low to high, not the other way around. Also, the module should have a reasonable amount of code associated with it.

Reference: Design Guidelines

Difficulty: medium

1. When putting individual components together for a structure chart you will need to identify the three basic kinds of processes in a process model. Identify these three kinds of models and describe each.

Answer

Three basic kinds of processes on a process model are afferent (input), central (main processing), and efferent (output). Afferent processes are processes that provide inputs into the system. Central processes perform critical functions in the operation of the system. Efferent processes deal with the system outputs.

Reference: Building the Structure Chart

Difficulty: medium