**Section A: Delegates:**

1. Define delegate.
2. Explain the declaration of a delegate.
3. Explain how to call different functions using a delegate.

* Delegate: an interface to transfer an action from one object to another.
* Links trigger to the action or event.
* Example: several other procedures with the same signature (Add, Subtract, Multiply) can reference the Calculate delegate.
* Single cast delegate: calls only one method
* Multicast delegate: calls more than one.
* Class Delegate definition: requires return type (specifies data type returned by function), identifier (name of delegate), and parameters (variables used in corresponding functions).
* AddressOf Keyword points to place in code where function is so delegate is used as a “function-pointer.”
* Delegate can be used to pick between functions with which to proceed. User can choose to add or subtract – delegates are used to point to either.
* Multicast delegates can point to many functions of the same signature. They refer to linked list (Invocation list) that holds locations of functions.
  + Combine ( ) adds a function to the list.
* Several procedures are chained when delegates are invoked.

**Section B: Events:**

1. Define events.
2. Explain the importance of events.
3. Declare an event and raise it.

* Event: trigger for an action to occur.
* Uses EITHER 1) user events (clicks and inputs), OR 2) program events.
* DECLARING EVENT:
  + Accessspecifier Event eventName(parameter)
* SPECIFY ACTION WHEN EVENT IS RAISED:
  + AddHandler eventName, AddressOf MyFunction
* RAISE EVENT:
  + RaiseEvent eventName
* When eventName is raised, MyFunction action takes place.

**Section C: Threads:**

1. Understand thread and multi-threading.
2. Create a new thread.
3. Suspend a thread.
4. Explain the thread life cycle.
5. Resume a thread.

* Thread: a chain of related tasks.
* A single process may have multiple threads that run parallel to each other.
* Multi-threading: term to describe multiple threads in a single process.
* One program 🡪 a process 🡪 multiple threads 🡪 that run parallel and do multiple tasks.
  + Ex: process: getting ready for school.
  + Thread: eat breakfast
  + Tasks: pour milk, make toast, prepare oatmeal.
* .NET framework has pre-defined namespace with classes and interfaces: System.Threading.
* Three stages in threading: (thread life cycle)
  + Creating, calling a thread.
  + Suspending
  + Resuming
* Thread life cycle (more detailed): describes states that threads go through:
  + Unstarted
  + Started
  + Running
  + Suspended/sleeping OR aborted
  + Resumed OR stopped.
  + Termination once program is over.
* DECLARE:
  + Dim thread as Thread = New Thread (AddressOf createThread)
  + Object is thread, AddressOf points to location of createThread delegate method.
* SUSPEND:

If thread.ThreadState = ThreadState.Running

Then  
    thread.Suspend()

End If

* RESUME:

If Myresumingthread.ThreadState = ThreadState.Suspended Then  
    Myresumingthread.Resume()  
End If