the screen, an outline will be drawn showing that the window will be resized to fill the screen. If the window is dragged to the right or left sides of the screen, an outline will be drawn showing that the window will be resized to fill half the screen on that side. If the window is left inside the borders of the screen it will resume the last size and shape it had before being docked to a side or top of the screen. Note that it is not until the user lets go of the mouse button that the screen will actually resize and lock itself into position.

Shake is a feature only available in versions of Windows 7 that are using the Aero theme. If a user clicks on the title bar of a window and shakes the mouse from side to side, all other windows will automatically minimize. Repeating the shake will restore all other windows to their original size and location.

Start Here

Hardware Requirements and System Hardware Support

Windows 7 is designed to provide a different look and feel depending on the version of the operating system and the capabilities of the system's hardware. A user or company only has to purchase the version and hardware they require. If desired, the operating system can be upgraded from one version to another if the need for enhanced features arises.

The degree to which a computer can be upgraded is determined by the manufacturer and its consideration for expandability and upgradeability. It is the consumer's responsibility to make sure the features of the computer they buy can be changed or expanded, or to determine that a nonupgradeable system will suit their needs.

Consumers have many computers and components to choose from. How do they know what will work with Windows 7? Microsoft has tried to simplify the choice by creating a testing program for computer hardware. Component manufacturers that want their product to be tested with Windows 7 can submit their solution to the **Windows Hardware Quality Labs** (**WHQL**). After it is thoroughly tested and deemed compatible, their solution will be publicly catalogued and recognized by Microsoft.

In the past, Microsoft maintained a **Hardware Compatibility List** (**HCL**) to keep track of which products would work with each operating system. The HCL was replaced with the Windows Catalog Web sites for products newer than Windows NT 4. Server class operating systems still have certified products listed on the Web at *www.windowsservercatalog.com*.

Legacy HCL lists and links to current compatibility Web sites, including Windows 7, can be found at *www.microsoft.com/whdc/hcl/*.

To help shoppers differentiate products, Microsoft has incorporated a logo program as part of the WHQL testing process. A vendor can display special "Compatible with Windows 7" logos on their hardware and software packaging to help the consumer make a better choice.

To add security and validity to the tested solutions, Microsoft will also include digital signatures as part of the hardware's drivers. Windows 7 will be able to recognize the digital signature and determine if it is safe to trust the driver. In a business environment, administrators can restrict the installation of drivers based on those digital signatures.

Even when a product passes initial testing and gets a Windows 7 logo, it may encounter difficulty and crash as the user adds patches, products, or makes configuration changes to their computer. Microsoft collects this information from the Windows Error Reporting (WER) tool built in to the operating system. When a crash occurs, a summary of what was happening on the computer is compiled and the user is asked if he or she wants to send this to Microsoft. The product's manufacturer can collect this data and find out more about their product stability.

Microsoft compiles a rating system for the manufacturer's drivers that scores how often people have problems with the driver and how many people it impacts. If the manufacturer maintains a low score for too long, Microsoft can revoke their logo status. To assist the manufacturer with distributing a patch, manufacturers who obtain logo status can distribute updated drivers through Windows Update—Microsoft's standard Web site for distributing patches and upgrades.

For WER tool users, the manufacturer can also send back a response to the user pointing them to the manufacturer's Web site for advice and updates.

Table 1-1 lists Window 7's minimum hardware requirements.

Table 1-1 Minimum hardware requirements for Windows 7

System Component	Recommendation
CPU	32- or 64-bit processor, 1 GHz or faster
System RAM	1 GB (2 GB for a computer with a 64-bit CPU)
Disk Space	16 GB for 32-bit editions, 20 GB for 64-bit editions
Video Card Drivers	DirectX 9 graphical processor and WDDM 1.0 (or higher)

Processor Support

Processing support in Windows 7 is designed for modern 32- and 64-bit processors. Processors that do not meet minimum recommendations may still be able to run Windows 7, but with some impact on features, performance, or stability. To enhance the performance of Windows 7, Microsoft has built-in support for several enhanced processor configurations.

Processes and Threads The actions performed by a **Central Processing Unit** (**CPU**) are defined by the instructions it is given. Programmers compile a list of instructions to build their applications. These instructions are typically grouped into units of code called **threads**. A thread is spawned, or started, by a process. The process itself is created by the applications and the operating system as they run. Threads and processes are common terms used to describe what the CPU is working on. To visualize what a thread and process represent, consider the following breakdown of an application.

A single application can be described by the tasks it must accomplish. For instance, we can describe the tasks a user is experiencing with a word processing application. The user will open a new document and type in text at the keyboard. The user wants the application to format a visual representation of the document, perform a spell check and grammar check, highlight errors it finds, and periodically save a copy of the text to disk. The word processing application in this case is the process. Formatting, spell check, and saving to disk are each executed by a different thread, or unit of code.

A single program that performs all of these tasks would be difficult to write and hard to maintain. The old DOS operating system ran applications that were essentially one big program with a single process running one thread at a time. To switch between threads, the DOS applications would typically need a trigger, such as the user pressing a key on the keyboard or a signal from the computer's clock. Typically, all of the application's code was written into a single file with a .COM or .EXE file extension. To switch between processes, a user would terminate one application and start another.

With the introduction of Windows, the idea of multitasking became popular. **Multitasking** gives the appearance that the computer is running multiple applications or processes at the same time. The operating system is switching from one thread to another very quickly, giving the illusion that all processes and their threads are running concurrently. In our word processing example, the user can see all those tasks happen at the same time while they type.

Applications designed to run in Windows run as one or more processes. A single **process** in Windows represents a collection of data, files, and instructions with a specific purpose while it is running. One or more application tasks can be assigned to a single process. In our example, the spelling and grammar check with suggested fixes can be part of one process; the auto-save to disk can be part of another.

Processes are typically described in Windows by the application they service, the user who launched them, and other attributes. The operating system uses its own processes to perform system actions such as managing files and network connections.

When a process executes a single task it will run a small block of code, a thread, for that task alone. The programmer decides what a single thread should do. Windows assigns that single thread to the CPU for execution.

For multitasking to work, a single task cannot take over the CPU for an extended period of time. In early versions of Windows, typically Windows 3.X, the applications and the operating system cooperated to share the CPU. This is called **cooperative multitasking**. The problem with this scheme is that a single task could take over the CPU and make it appear that the computer has stopped responding.

Preemptive multitasking was introduced as an improvement over cooperative multitasking in later versions of Windows and is used by Windows 7. This allows a single process to be interrupted by another process, even if the first process has not completed. To control the interruptions, Windows uses a system of priority levels and time windows to control scheduling of the processes and threads.

Each thread is given a window of time to execute in before the operating system checks to see if the CPU should switch to another thread. If the thread has not finished its task, it must wait for its next turn. The time window a thread is allowed to run in is known as a **quantum**. The thread can be preempted by another thread before its quantum is over—even before it has started processing.

To help determine which thread gets to go next, and which threads are allowed to preempt others, the threads and processes are assigned a priority level. The higher the priority level, the greater the chance that the process will preempt the current thread or get the next quantum. If there are no threads that are ready to run, there is an operating system process (the Idle Process) always ready to run.

If a thread is not finished running, perhaps because it had to wait or it was preempted, it is typically restarted on the same processor that previously ran it. This is known as **processor affinity**, where the thread is restricted to which CPU can run it.

When multiple processes and threads are running, it doesn't make sense for a programmer to write all of the instructions for an application into a single file. Windows programs are usually written in a modular nature, with different files holding different pieces of the application. Code modules are saved in **Dynamic Link Library files** (**DLLs**). Code modules in the DLLs can be shared between applications. Updates to applications can replace individual DLLs instead of the entire application.

Activity 1-4: Switching between Applications

Time Required: 5 minutes
Objective: Observe how to switch between running applications

Description: In this activity, you will start multiple applications and use the application switcher feature of Windows 7 to quickly change which application is in the foreground.

1. If necessary, start your computer and log on as **User*x*,** where ***x*** is the student number assigned by your instructor.
2. Start **Notepad** and enter some random text.
3. Start **Internet Explorer** and leave it at the starting page.
4. Hold down the **Alt** key and press the **Tab** key once. Do not release the **Alt** key.
5. Notice that each running application has its active content displayed in a miniature preview window.
6. Press the **Tab** key repeatedly to cycle the highlighted box from one application to the next.

7. Press the **Tab** key until Notepad is highlighted. Release the **Alt** key.
8. Notice that the **Notepad** application becomes the foreground application.

The remainder of the activity requires the Windows Aero interface to be active on your computer.

9. Hold down the **Windows** key on the keyboard and press **Tab** once. Do not release the **Windows** key.
10. Notice that each running application is displayed in a 3D preview window.
11. Press the **Tab** key repeatedly to cycle the 3D windows. Press the **Tab** key until the Notepad window is the top window. Release the **Windows** key.
12. (Optional step). If your mouse has a scroll-wheel control you can test the following. Press the **Windows** and **Tab** keys as in Step 9. While you are holding down the **Windows** key scroll the wheel on the mouse to cycle through the 3D preview windows. Release the **Windows** key when Notepad is the top window.
13. Close all applications without saving any changes and log off.

Stop Here :)

Activity 1-5: Working with Task Manager

Time Required: 20 minutes
Objective: Observe how to start and stop applications with Task Manager

Description: The operating system is constantly starting and terminating processes as required. The Task Manager tool enables users to monitor and manage this activity. In this activity, you will start Task Manager and use it to start and stop applications and processes. You will see how Task Manager can filter which processes are shown to the user and how to sort the list of running processes.

1. If necessary, start your computer and log on.
2. Start **Notepad** and enter some random text.
3. Start **Internet Explorer** and leave it at the starting page.
4. Right-click the system clock at the bottom right of the screen.
5. Select **Start Task Manager** from the pop-up menu.
6. Notice that if this is the first time Windows Task Manager is opened the **Applications** tab is initially selected. Windows Task Manager will remember the last tab that was opened and make it the default to view the next time the program is launched. If the Applications tab is not already selected click it to select it.
7. Highlight the **Untitled—Notepad** application and then click on the **End Task** button. Note that you are prompted by notepad to save your work. A new window will appear called End Program – Untitled – Notepad. Click the **End Now** button to force Notepad to close.
8. Notice that Notepad is no longer running.
9. In the Windows Task Manager window, click the **New Task . . .** button.
10. In the Open field, enter **notepad** and click on the **OK** button.
11. Notice that Notepad has started in the background and is now listed on the Applications tab. By default the Task Manager window is always displayed on top of all running application windows.
12. Click the **Processes** tab in Windows Task Manager.