Ex.No:1 Date:	**INSTALLATION OF WINDOWS OPERATING SYSTEM**

AIM

To install windows operating system

PROCEDURE:

Windows XP Installation

It is important to understand that this guide was specifically designed for a lab environment. There are a lot of operating system vulnerabilities that are intentionally left unpatched in theseinstallation steps. This is intentionally done to give you the best results when completing the labs and tutorials in this book. If you are interested, a great reference for building a Windows XP Professional box that is secure enough for a production environment is *Windows XP Security: Step By Step* by SANS.

To create a properly configured laptop for the Security Essentials Boot Camp, follow the detailed steps in this document—from the initial setup screen to the final login. This guide was designed for use on a system that doesn't already have a Windows platform installed on it. If your machine does not have a blank hard drive, some of the screens you see at the beginning ofthe installation may be different from what you see in this chapter. If different screens appear, it is important that you always choose the option to *replace*, or *overwrite*. Do not choose to upgrade. The Windows install should also be placed in the default c:\windows directory.

Creating Boot Disks

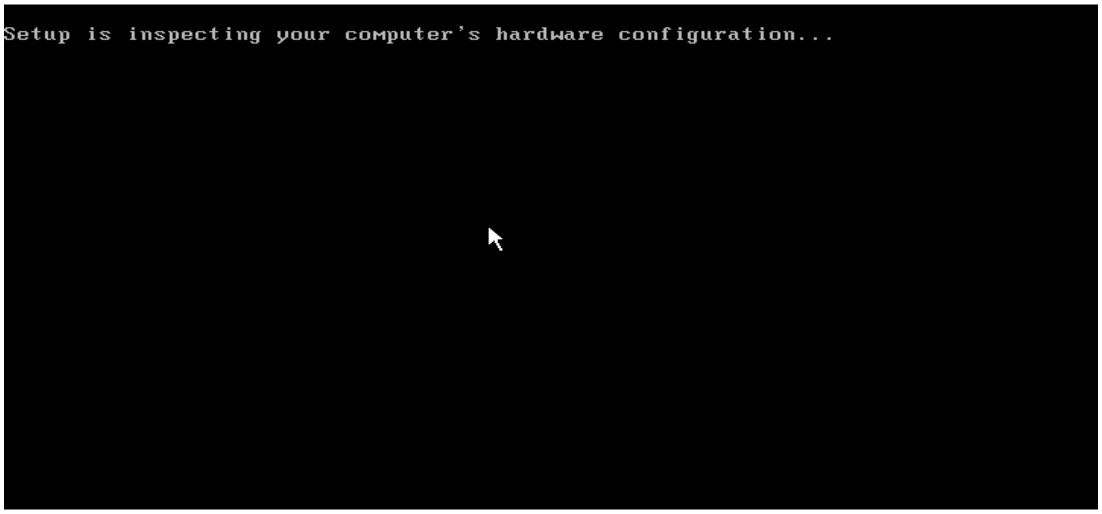
If your system does not support the capability to boot off of a CD-ROM, you can use the WindowsXP boot disk to boot. If you do not have a set of the four disks, you need to use a machine that already has Windows XP Professional installed on it. The following steps show you how to createthe four boot disks:

1. Label four blank, formatted, 3.5-inch, 1.44-MB floppy disks as: Setup Disk One, Setup Disk Two, Setup Disk Three, and Setup Disk Four.
2. Insert Setup Disk One into the floppy disk drive of a Windows or DOS system.
3. Insert the Windows XP CD-ROM into the CD-ROM drive.
4. Click Start, and then click Run.
5. In the Open box, type D:\bootdisk\makeboot a: (where D: is the drive letter assigned to your CD-ROM drive), and then click OK.
6. Follow the screen prompts.
7. After you have completed the screen prompt requests, insert Setup Disk One into the floppy disk drive of the lab PC and power the PC on.

Booting from the CD-ROM

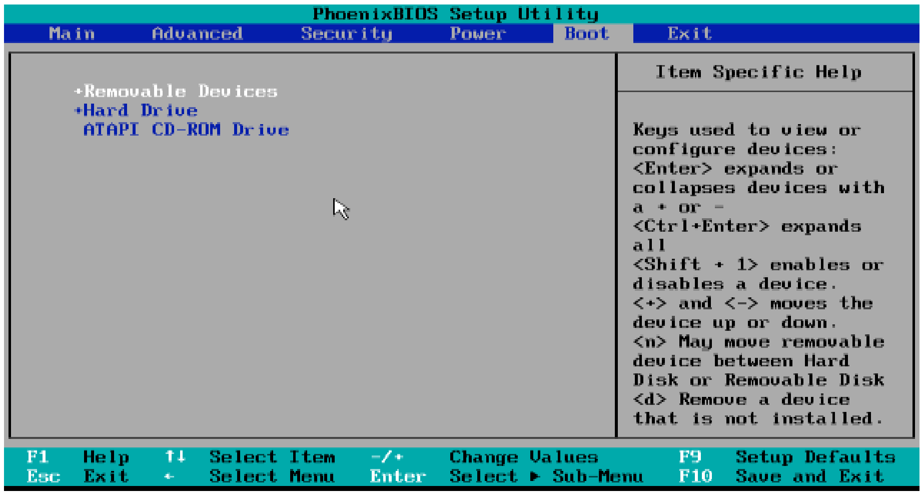
If your system supports booting off of the CD-ROM, you do not need to use the disks previously discussed. Instead, follow these steps:

1. Simply start by placing the Windows XP CD-ROM into your CD tray and power on your machine. The first non-blank screen you should see the one shown is following illustration.



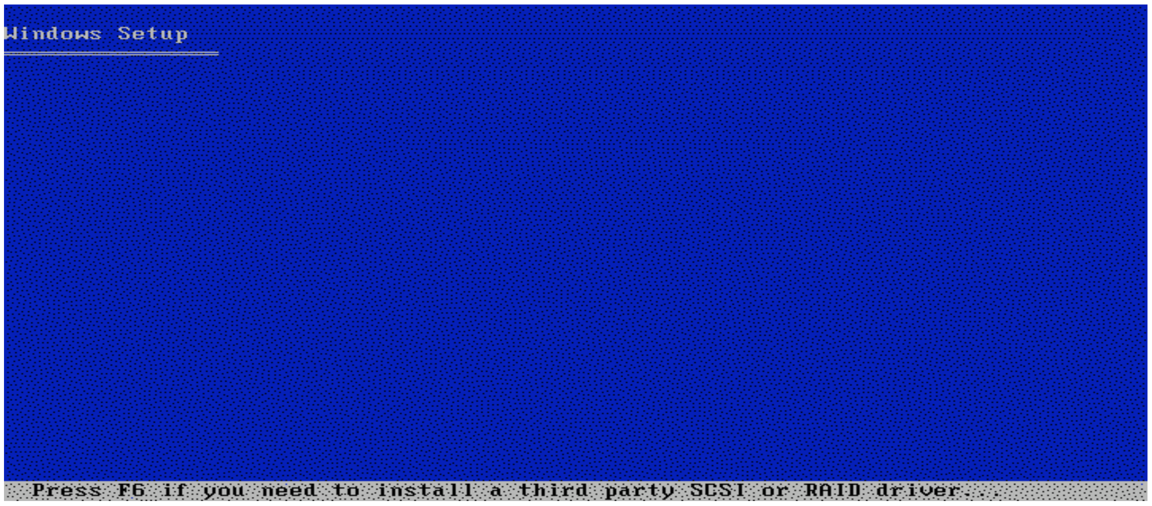
```
Setup is inspecting your computer's hardware configuration...
```

2. If the previous screen does not appear, reboot your machine and open up the BIOS. You need to make the system boot to the CD-ROM first. The following screen is one of several different BIOS you could have on your system. You need to navigate to a screen that allows you to change the Boot Order. This is where you tell it to boot off of the CD-ROM.



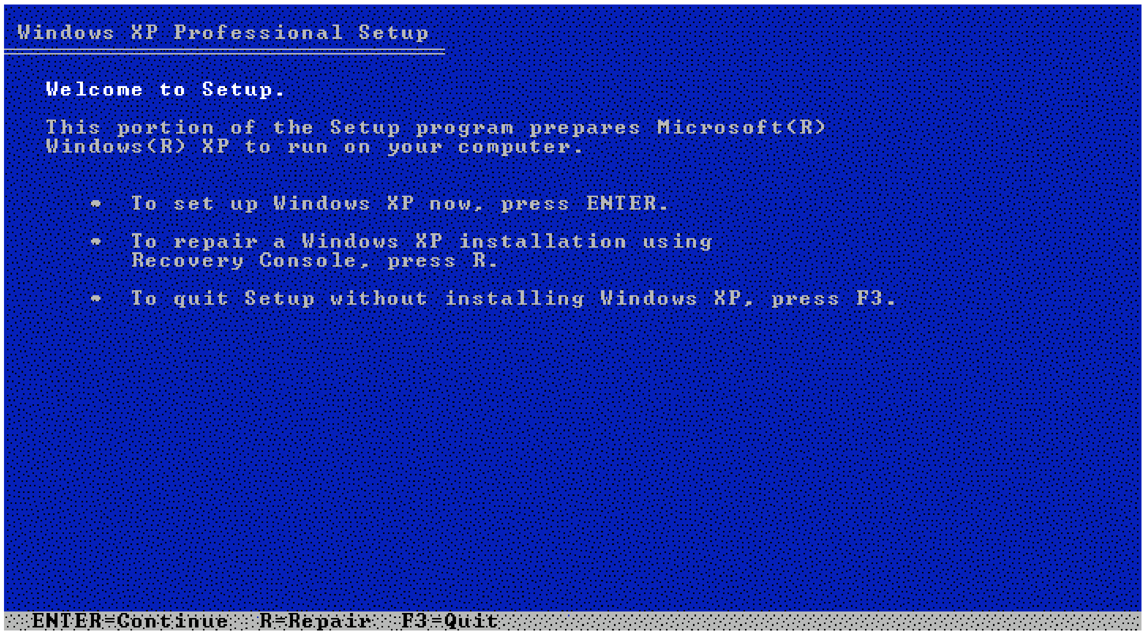
```
                        PhoenixBIOS Setup Utility
    Main     Advanced     Security     Power     Boot     Exit

       +Removable Devices                         Item Specific Help
       +Hard Drive
        ATAPI CD-ROM Drive                      Keys used to view or
                                                configure devices:
                                                <Enter> expands or
                                                collapses devices with
                                                a + or -
                                                <Ctrl+Enter> expands
                                                all
                                                <Shift + 1> enables or
                                                disables a device.
                                                <+> and <-> moves the
                                                device up or down.
                                                <n> May move removable
                                                device between Hard
                                                Disk or Removable Disk
                                                <d> Remove a device
                                                that is not installed.

 F1   Help   ↑↓  Select Item   -/+    Change Values            F9   Setup Defaults
 Esc  Exit   ↔   Select Menu   Enter  Select ► Sub-Menu        F10  Save and Exit
```

3. Now your system should boot off of the CD-ROM. After a period of time (typically 30-45 seconds), the following screen appears. Because we are doing an initial install, you only need to press Enter to continue.



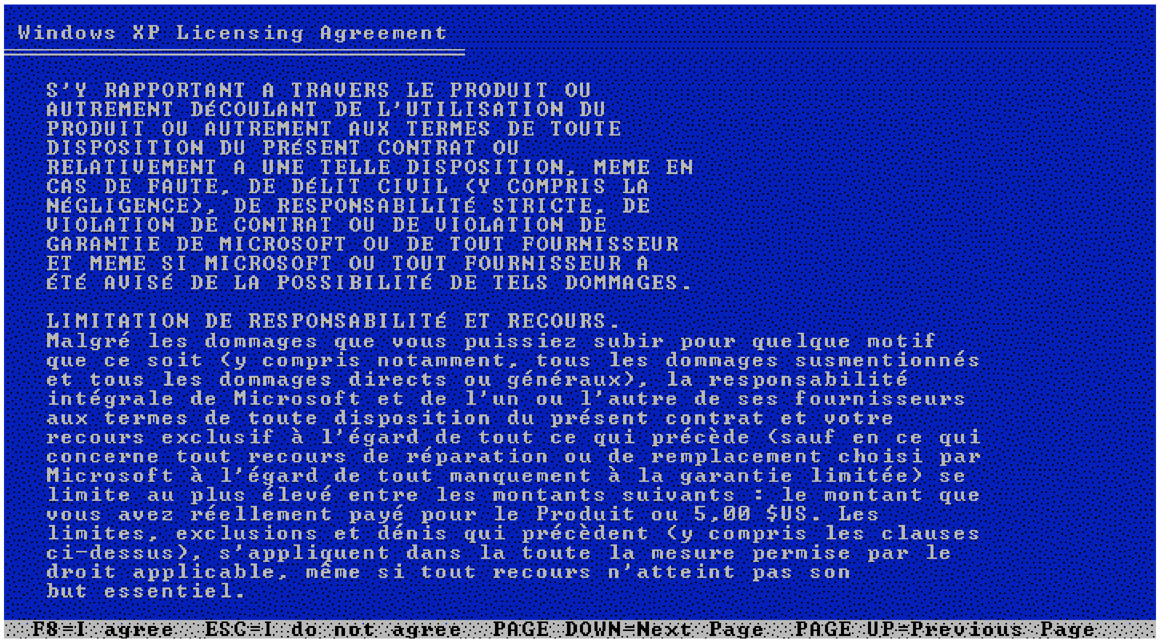
```
Windows Setup

 Press F6 if you need to install a third party SCSI or RAID driver...
```

4. Hit **Enter** at the next screen to continue installation.



```
Windows XP Professional Setup

   Welcome to Setup.

   This portion of the Setup program prepares Microsoft(R)
   Windows(R) XP to run on your computer.

      •  To set up Windows XP now, press ENTER.

      •  To repair a Windows XP installation using
         Recovery Console, press R.

      •  To quit Setup without installing Windows XP, press F3.

  ENTER=Continue  R=Repair  F3=Quit
```

5. The Microsoft Windows XP Licensing Agreement appears next, as shown in the following screen. It is important that you read and understand this agreement before continuing with the installation. After you have read and agreed to the contents of the license, press F8 to continue.



```
Windows XP Licensing Agreement

   S'Y RAPPORTANT A TRAVERS LE PRODUIT OU
   AUTREMENT DÉCOULANT DE L'UTILISATION DU
   PRODUIT OU AUTREMENT AUX TERMES DE TOUTE
   DISPOSITION DU PRÉSENT CONTRAT OU
   RELATIVEMENT A UNE TELLE DISPOSITION, MEME EN
   CAS DE FAUTE, DE DÉLIT CIVIL (Y COMPRIS LA
   NÉGLIGENCE), DE RESPONSABILITÉ STRICTE, DE
   VIOLATION DE CONTRAT OU DE VIOLATION DE
   GARANTIE DE MICROSOFT OU DE TOUT FOURNISSEUR
   ET MEME SI MICROSOFT OU TOUT FOURNISSEUR A
   ÉTÉ AVISÉ DE LA POSSIBILITÉ DE TELS DOMMAGES.

   LIMITATION DE RESPONSABILITÉ ET RECOURS.
   Malgré les dommages que vous puissiez subir pour quelque motif
   que ce soit (y compris notamment, tous les dommages susmentionnés
   et tous les dommages directs ou généraux), la responsabilité
   intégrale de Microsoft et de l'un ou l'autre de ses fournisseurs
   aux termes de toute disposition du présent contrat et votre
   recours exclusif à l'égard de tout ce qui précède (sauf en ce qui
   concerne tout recours de réparation ou de remplacement choisi par
   Microsoft à l'égard de tout manquement à la garantie limitée) se
   limite au plus élevé entre les montants suivants : le montant que
   vous avez réellement payé pour le Produit ou 5,00 $US. Les
   limites, exclusions et dénis qui précèdent (y compris les clauses
   ci-dessus), s'appliquent dans la toute la mesure permise par le
   droit applicable, même si tout recours n'atteint pas son
   but essentiel.
  F8=I agree   ESC=I do not agree   PAGE DOWN=Next Page   PAGE UP=Previous Page
```

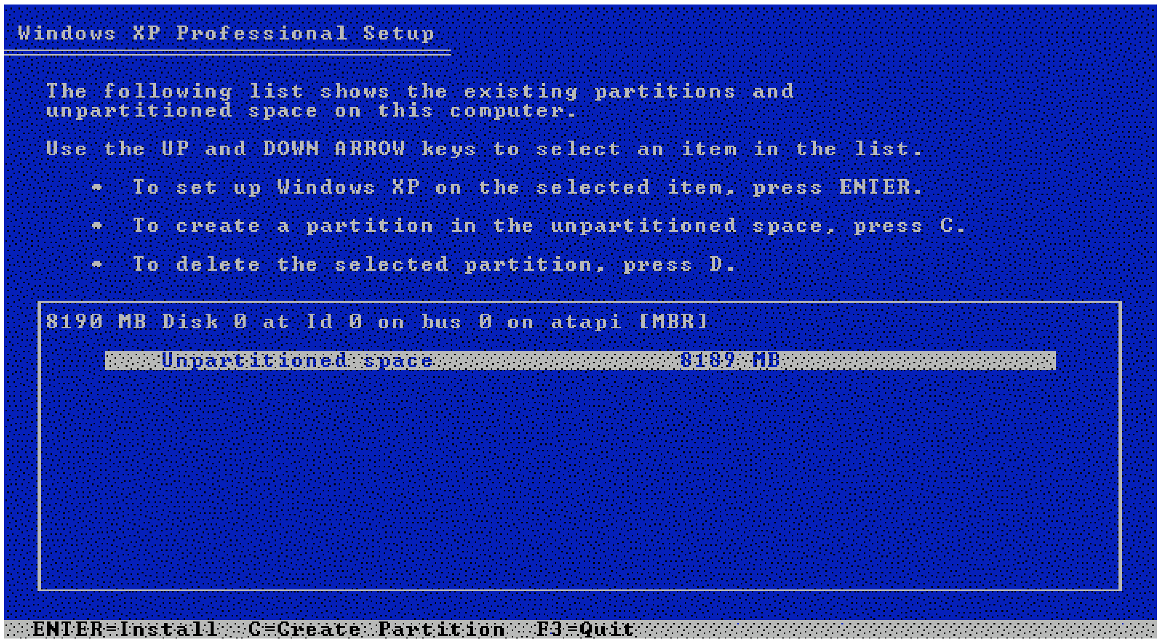
Defining Drive Partitions

You now need to define the drive partitions. Defining your drive partitions is used instead ofFDISK. When defining your drive partitions, it is extremely important that you leave enough space for your Linux partition! Following are the steps:

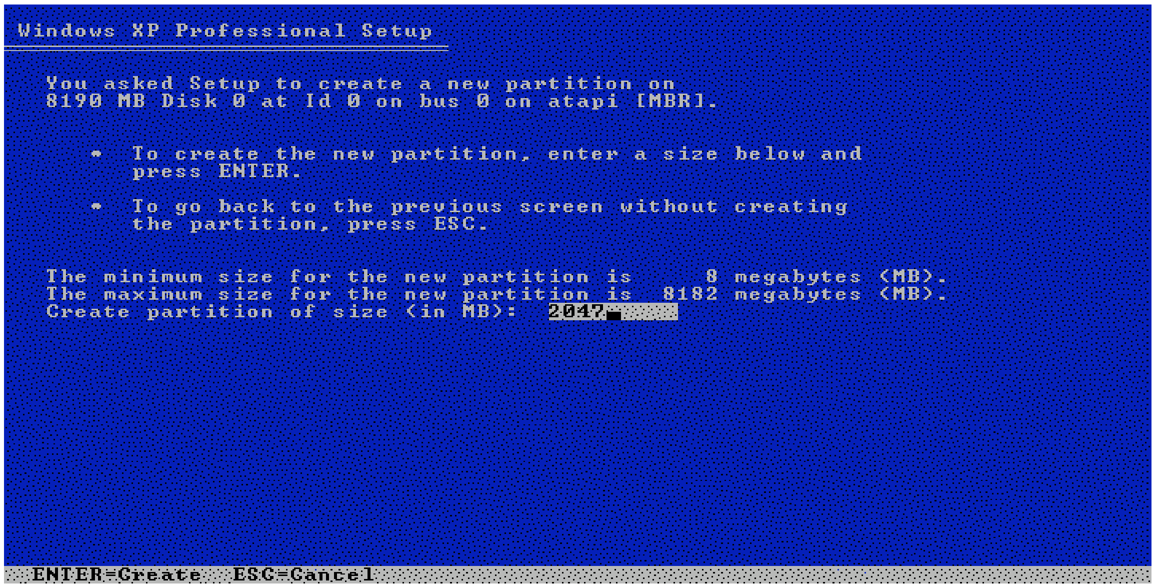
1. Press **C** to create a partition for your Windows install.

2. You need a minimum of 2Gb of space for each of your operating systems. When you are prompted for the size of the partition, enter a number that isequal to 50 percent of your available hard drive space. Then, highlight the partition, which should be labeled **Unpartitioned space** (see the following illustration)**,** and press **C**.

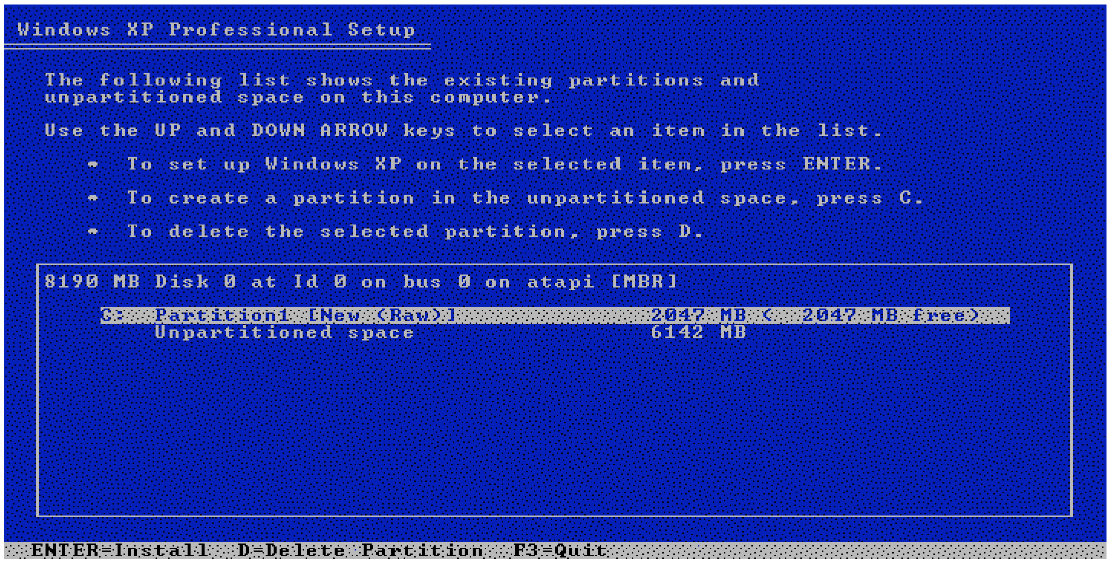
Note: If partitions already exist they should be deleted. However you should realize that this willpermanently remove any data that is currently on your system.


```
Windows XP Professional Setup

  The following list shows the existing partitions and
  unpartitioned space on this computer.

  Use the UP and DOWN ARROW keys to select an item in the list.

      •  To set up Windows XP on the selected item, press ENTER.

      •  To create a partition in the unpartitioned space, press C.

      •  To delete the selected partition, press D.

  8190 MB Disk 0 at Id 0 on bus 0 on atapi [MBR]

        Unpartitioned space                    8189 MB

  ENTER=Install  C=Create Partition  F3=Quit
```

Now create your new partition to be at least 2 Gb. In the provided space type **2047** and press **Enter**


```
Windows XP Professional Setup

  You asked Setup to create a new partition on
  8190 MB Disk 0 at Id 0 on bus 0 on atapi [MBR].

     •  To create the new partition, enter a size below and
        press ENTER.

     •  To go back to the previous screen without creating
        the partition, press ESC.

  The minimum size for the new partition is     8 megabytes (MB).
  The maximum size for the new partition is  8182 megabytes (MB).
  Create partition of size (in MB):   2047

  ENTER=Create   ESC=Cancel
```

4.You should now see two partitions. Verify that the new 2047 partition is highlighted and press **Enter**.

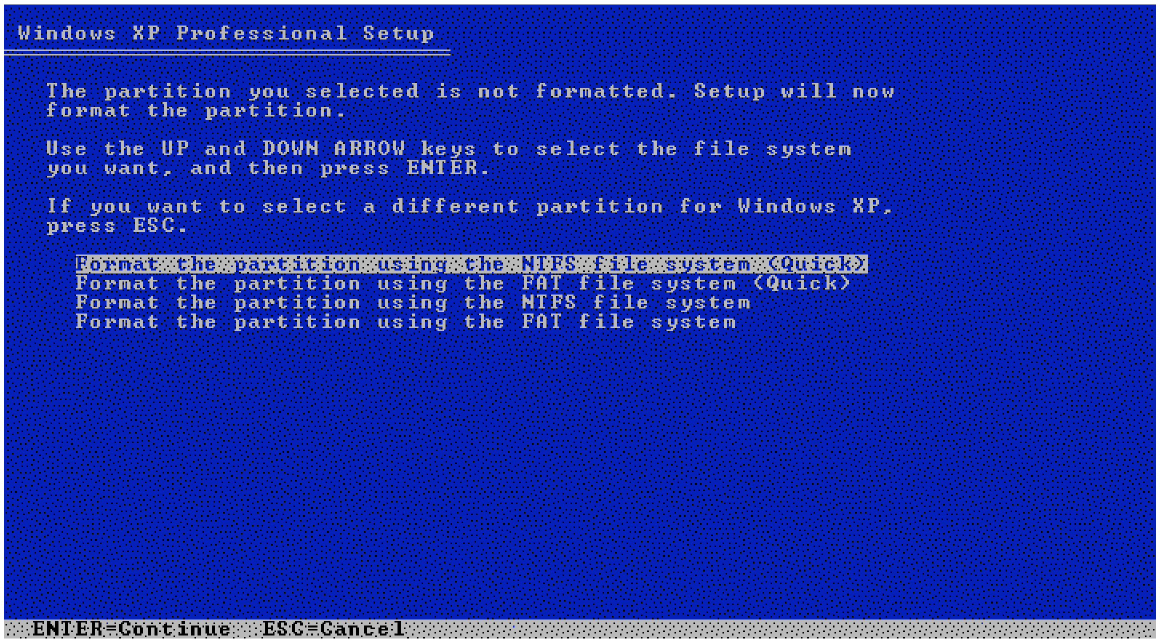


```
Windows XP Professional Setup

  The following list shows the existing partitions and
  unpartitioned space on this computer.

  Use the UP and DOWN ARROW keys to select an item in the list.

     •  To set up Windows XP on the selected item, press ENTER.

     •  To create a partition in the unpartitioned space, press C.

     •  To delete the selected partition, press D.

  8190 MB Disk 0 at Id 0 on bus 0 on atapi [MBR]

      C:  Partition1 [New (Raw)]              2047 MB (  2047 MB free)
          Unpartitioned space                 6142 MB

 ENTER=Install  D=Delete Partition  F3=Quit
```

Formatting Drive Partitions

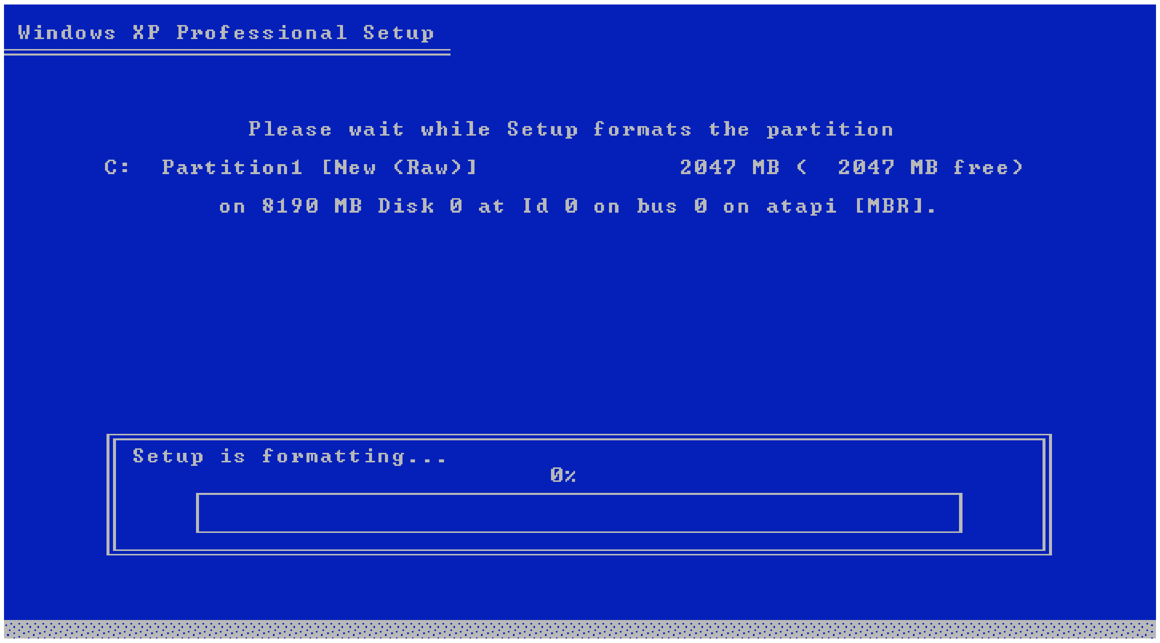
The next step is to format your partition. For security reasons, you should format your partitions using NTFS. NTFS is a Windows partition type that allows you to assign permissions at the folder level. This level of granularity is not the same for FAT partitions. NTFS also allows for lager partition sizes compared to the 2Gb limit that comes with FAT16.The steps for formatting your partition follow:

1. Highlight the NTFS <Quick> partition option as shown in the followingscreen, and press **Enter**.



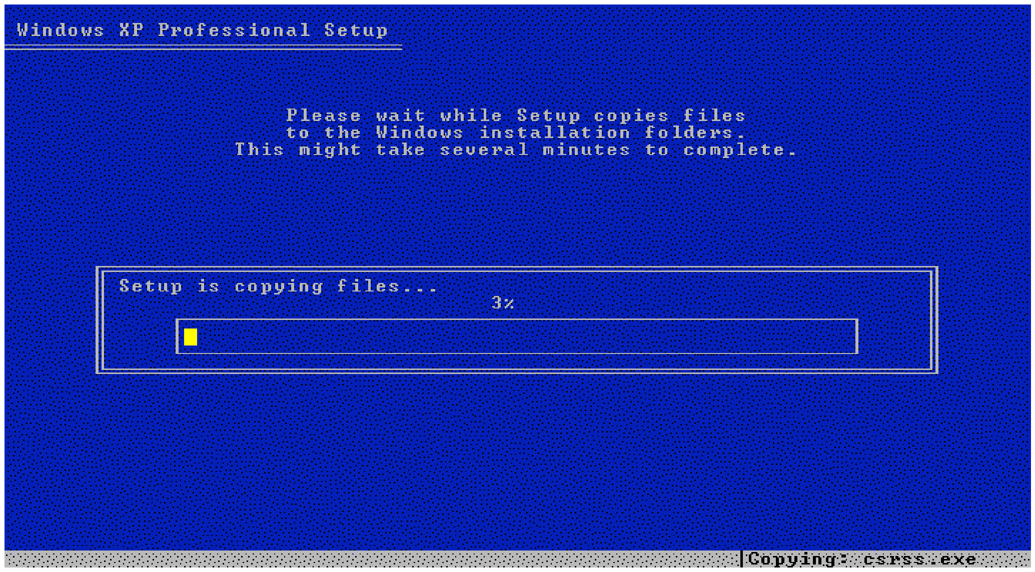
```
Windows XP Professional Setup

  The partition you selected is not formatted. Setup will now
  format the partition.

  Use the UP and DOWN ARROW keys to select the file system
  you want, and then press ENTER.

  If you want to select a different partition for Windows XP,
  press ESC.

    Format the partition using the NTFS file system (Quick)
    Format the partition using the FAT file system (Quick)
    Format the partition using the NTFS file system
    Format the partition using the FAT file system

 ENTER=Continue  ESC=Cancel
```

2.After you press **Enter**, the system formats the partition, as shown in the following screens. Depending on the size of the partition, this step can take from 5 minutes to an hour. This is a great time to refill your caffeine-laced beverage of choice. (You may need it because you have a long wayto go.)



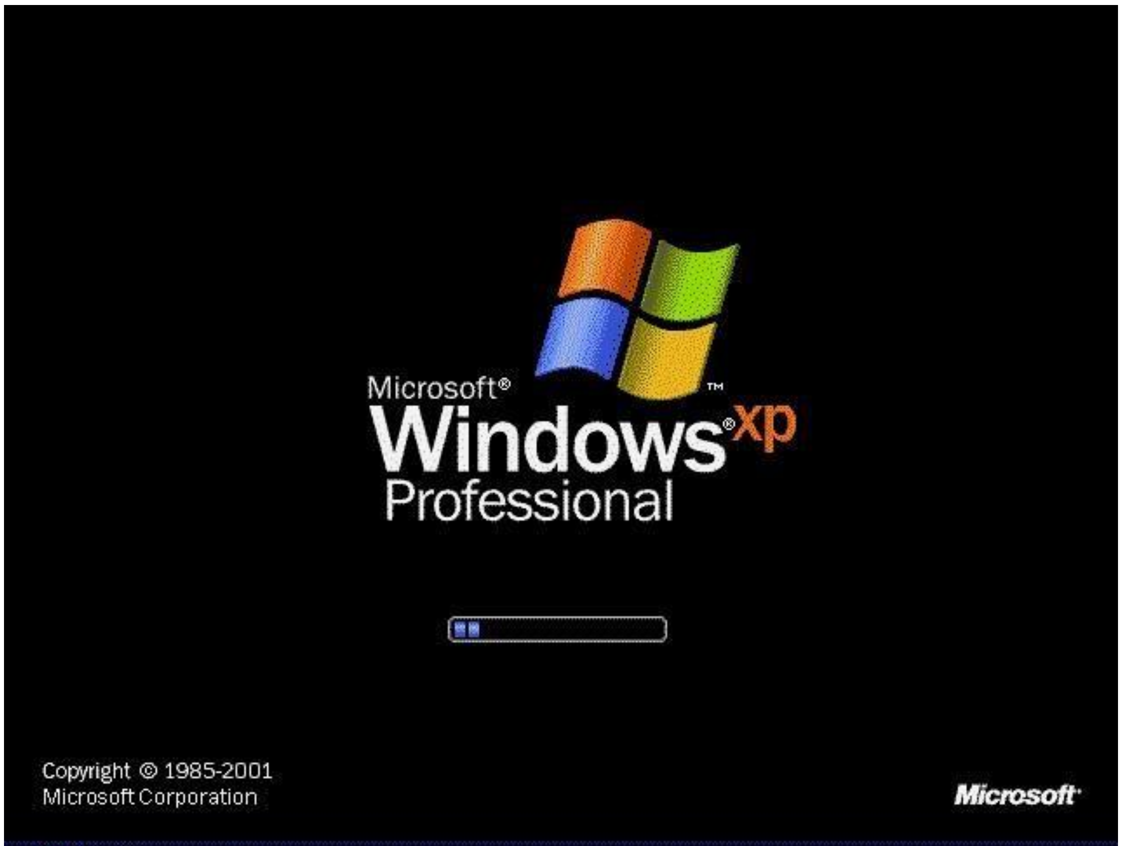
```
Windows XP Professional Setup

                 Please wait while Setup formats the partition

     C:  Partition1 [New (Raw)]               2047 MB (  2047 MB free)

            on 8190 MB Disk 0 at Id 0 on bus 0 on atapi [MBR].

   Setup is formatting...
                                 0%
```

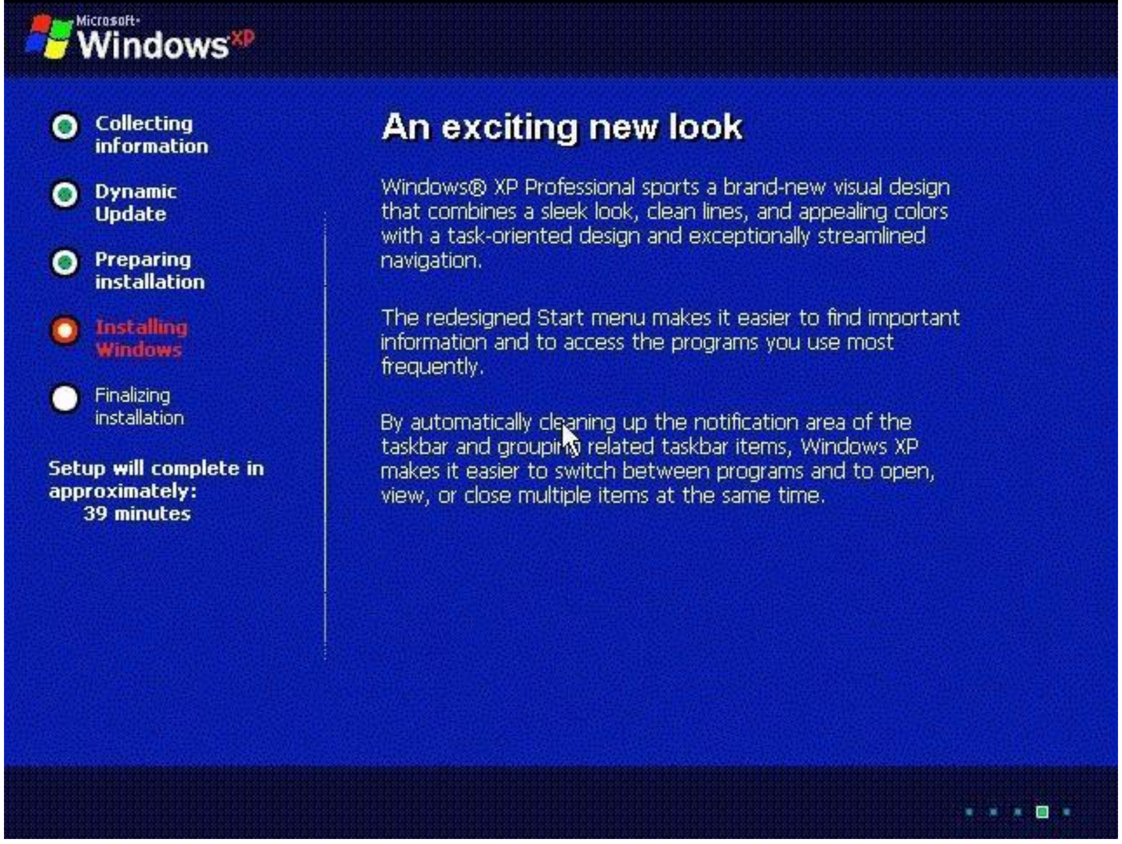
Since this will take a while you should just wait while this process continues.



```
Windows XP Professional Setup

                  Please wait while Setup copies files
                  to the Windows installation folders.
                This might take several minutes to complete.

   Setup is copying files...
                               3%

                                                  Copying: csrss.exe
```

When you return to your machine, you may see one of the following screens. Don't be alarmed. The system has completed the formatting process and has automatically rebooted.After this occurs, you have to answer the remaining install questions.





Setup will complete in approximately:
39 minutes

An exciting new look

Windows® XP Professional sports a brand-new visual design that combines a sleek look, clean lines, and appealing colors with a task-oriented design and exceptionally streamlined navigation.

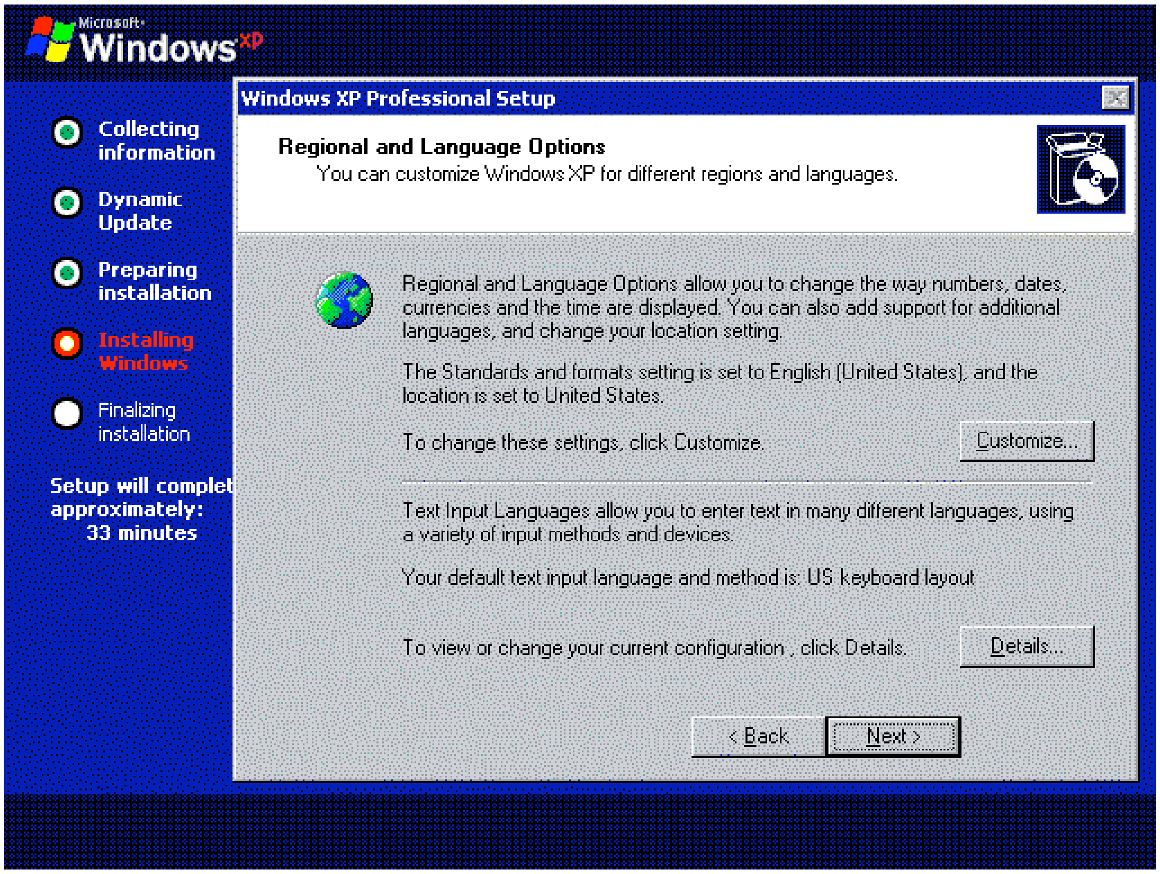
The redesigned Start menu makes it easier to find important information and to access the programs you use most frequently.

By automatically cleaning up the notification area of the taskbar and grouping related taskbar items, Windows XP makes it easier to switch between programs and to open, view, or close multiple items at the same time.

Customizing Your System

Now Windows presents a series of questions, which, when answered, customize your system.The following steps walk you through the process of customizing your system:

1. Typically, you only need to make changes during the next step (see the following screen) if you are located outside of the United States or if youuse a non-standard keyboard. If you are in the United States and you are using a standard QWERTY keyboard, press the **Next** button. If you are located outside of the United States, you should change your locale settings.


Microsoft Windows XP
Collecting information
Dynamic Update
Preparing installation
Installing Windows
Finalizing installation
Setup will complet approximately: 33 minutes
Windows XP Professional Setup
Regional and Language Options
You can customize Windows XP for different regions and languages.
Customize...
Details...
< Back
Next >

Regional and Language Options allow you to change the way numbers, dates, currencies and the time are displayed. You can also add support for additional languages, and change your location setting.

The Standards and formats setting is set to English (United States), and the location is set to United States.

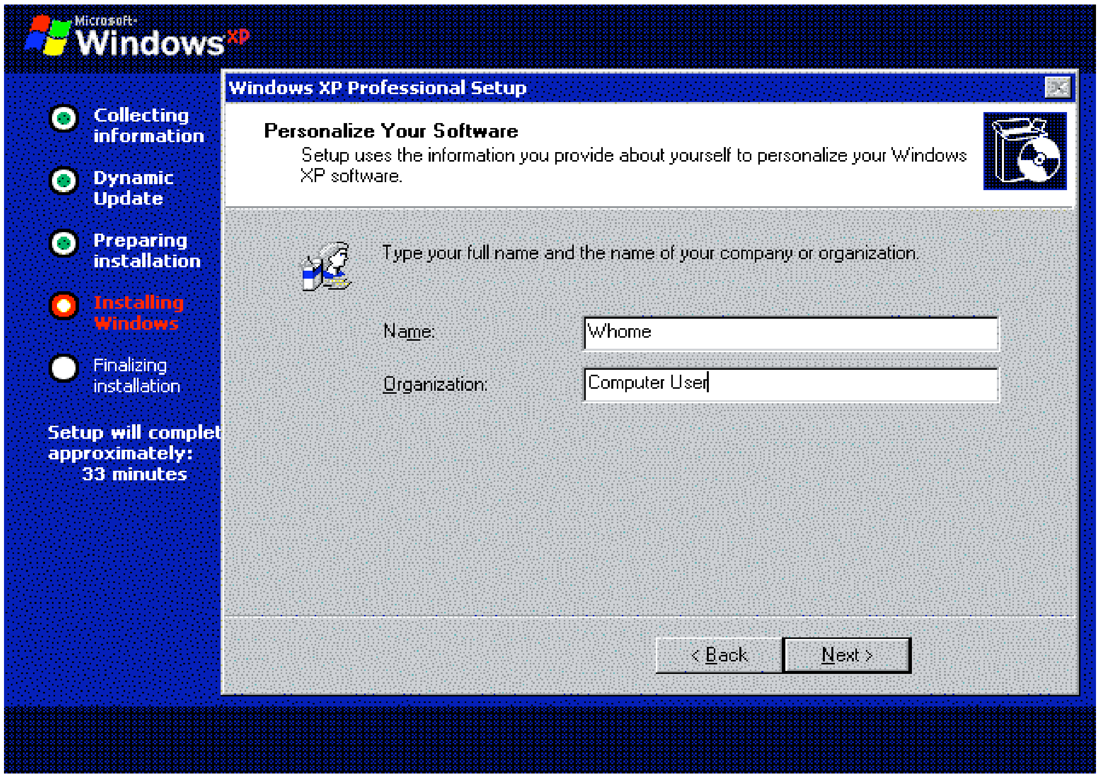
To change these settings, click Customize.

Text Input Languages allow you to enter text in many different languages, using a variety of input methods and devices.

Your default text input language and method is: US keyboard layout

To view or change your current configuration , click Details.

2. Enter your name and the organization you work for in the **Name** and **Organization** fields. For the purposes of this course, have some fun making up fictional names. Click the **Next** button when you are done.


Windows XP Professional Setup

Personalize Your Software

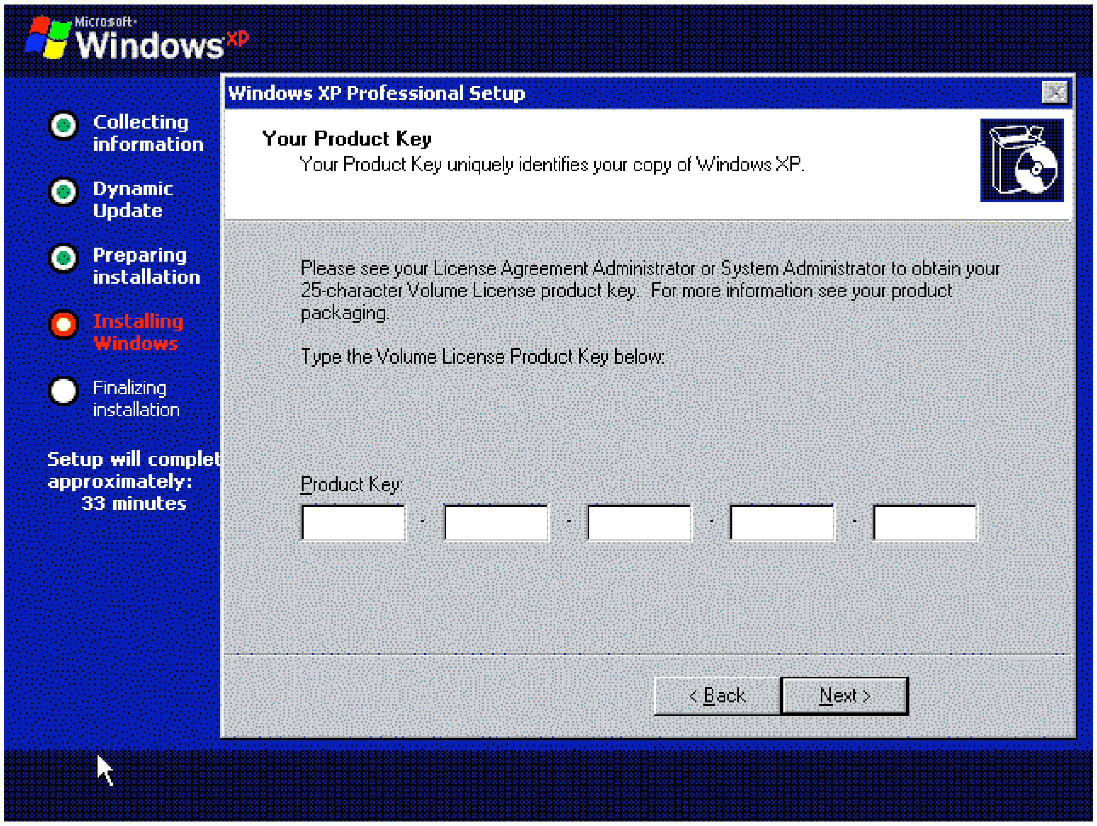
Setup uses the information you provide about yourself to personalize your Windows XP software.

Type your full name and the name of your company or organization.

Name: Whome

Organization: Computer User

3. In the next screen, enter the Product Key number that came with your software (find it on your CD). If you make a mistake when you enter thekey, you receive an **Invalid Key** message and the system gives you another opportunity to enter it. Once you enter in the valid key, press the**Enter** key.


Windows XP Professional Setup

Your Product Key

Your Product Key uniquely identifies your copy of Windows XP.

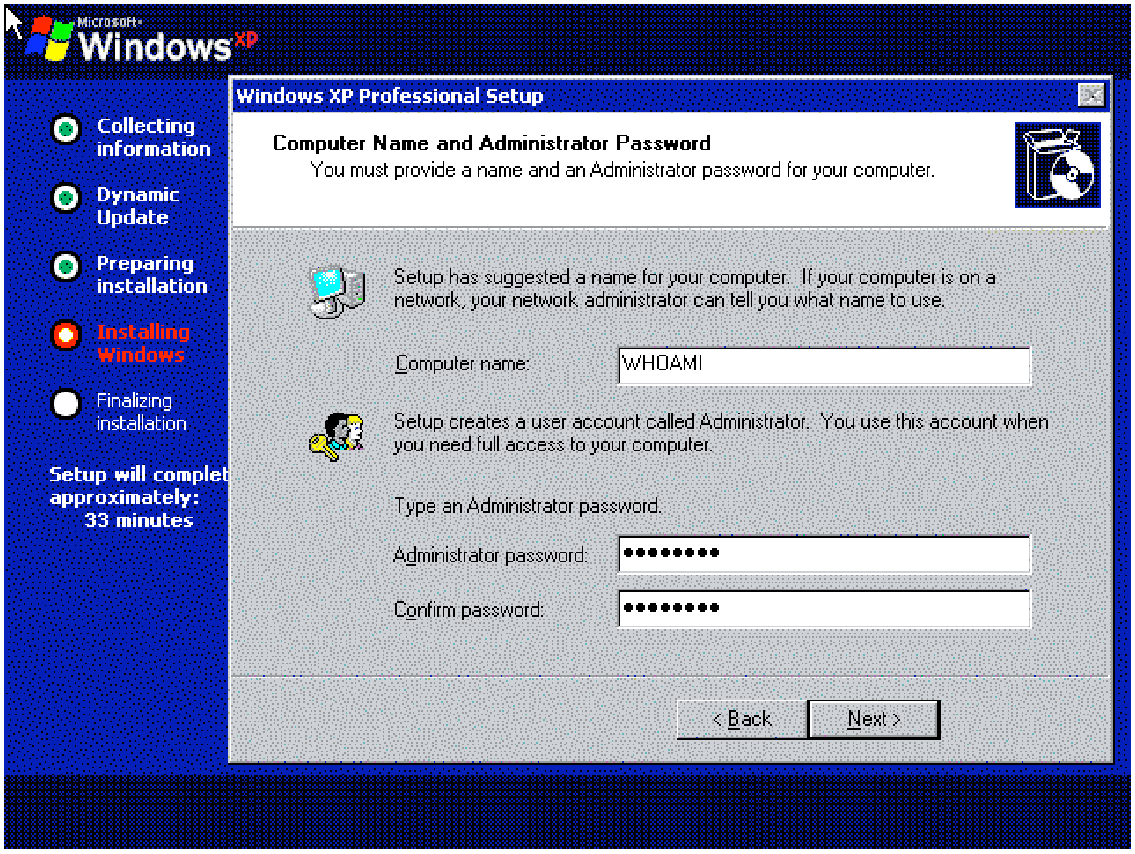
Please see your License Agreement Administrator or System Administrator to obtain your 25-character Volume License product key. For more information see your product packaging.

Type the Volume License Product Key below:

Product Key:

4. Now enter a name in the **Computer name** field to name your computer. Ifyou are part of a corporation's domain, you need to follow your corporation's guidelines for naming systems. For our purposes, name yourmachine whatever you desire. Then, type in a password in the **Administrator password** field. You also need to confirm the password, as shown in the following screen. Then, click the **Next** button.

Warning: A common mistake many administrators make at this stage is to leavethe **Administrator password** field blank. It is highly advisable that you enter a password that matches your company's password policy for local passwords.
You don't want to forget to change the password after you have completed theinstallation. Also, make sure you remember this password. You will need it to login.



Computer Name and Administrator Password
You must provide a name and an Administrator password for your computer.

Setup has suggested a name for your computer. If your computer is on a network, your network administrator can tell you what name to use.

Computer name: WHOAMI

Setup creates a user account called Administrator. You use this account when you need full access to your computer.

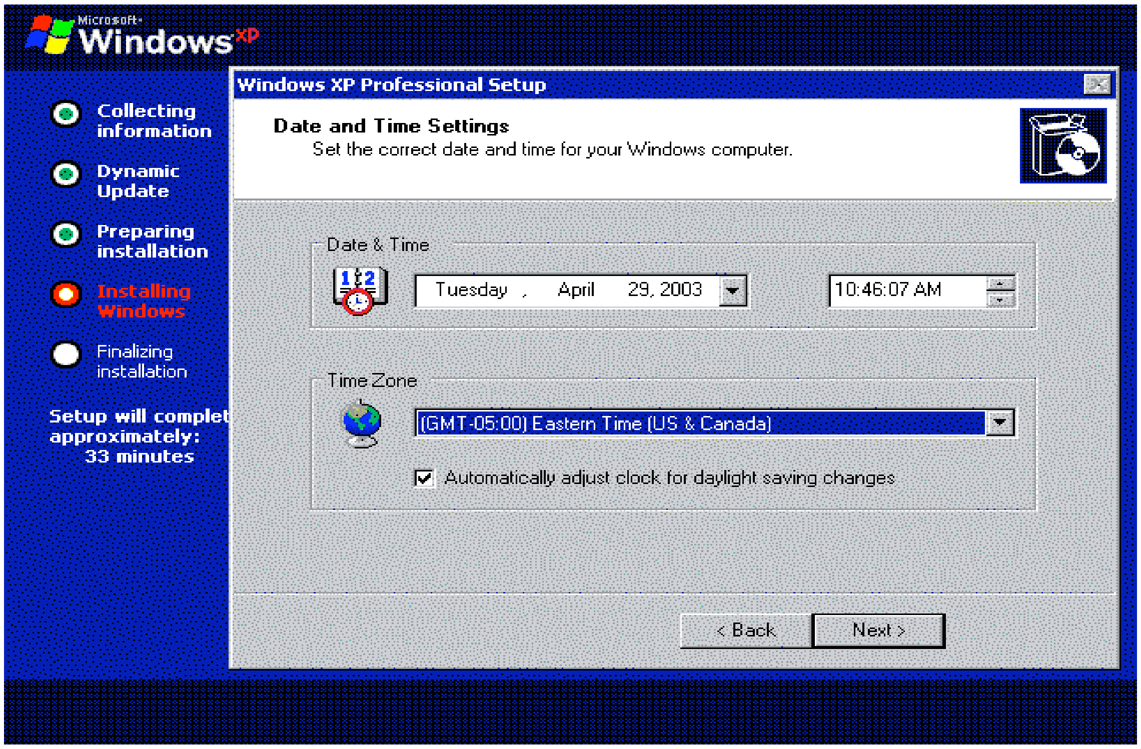
Type an Administrator password.

Administrator password: ••••••••

Confirm password: ••••••••

Note: Depending on your configuration, you might receive the Modem Dialing Information Screen. Just cancel out of this or click Next to get to the next screen.

6. In the screen that appears, enter the current time, and then fill in the **Date** field and **Time Zone** field. Click **Next**.



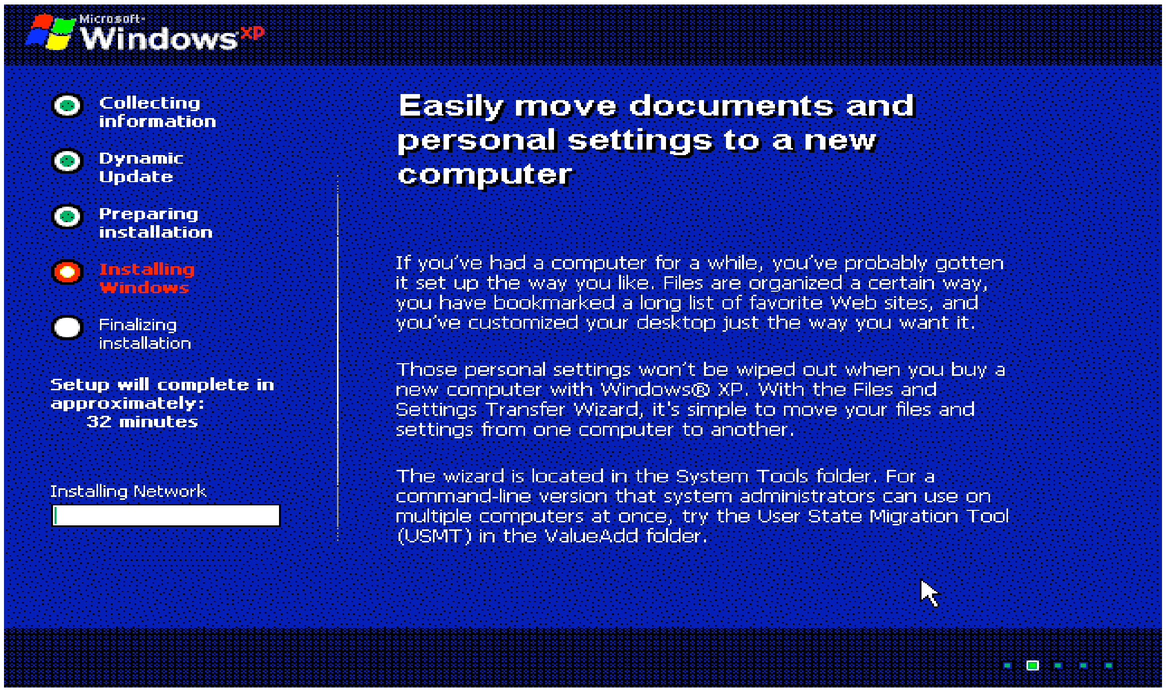
Date and Time Settings
Set the correct date and time for your Windows computer.

Date & Time: Tuesday , April 29, 2003 — 10:46:07 AM

Time Zone: (GMT-05:00) Eastern Time (US & Canada)

☑ Automatically adjust clock for daylight saving changes

6.After you make the previous configurations, the system installs yournetworking components, as shown in the following screen.



Easily move documents and personal settings to a new computer

If you've had a computer for a while, you've probably gotten it set up the way you like. Files are organized a certain way, you have bookmarked a long list of favorite Web sites, and you've customized your desktop just the way you want it.

Those personal settings won't be wiped out when you buy a new computer with Windows® XP. With the Files and Settings Transfer Wizard, it's simple to move your files and settings from one computer to another.

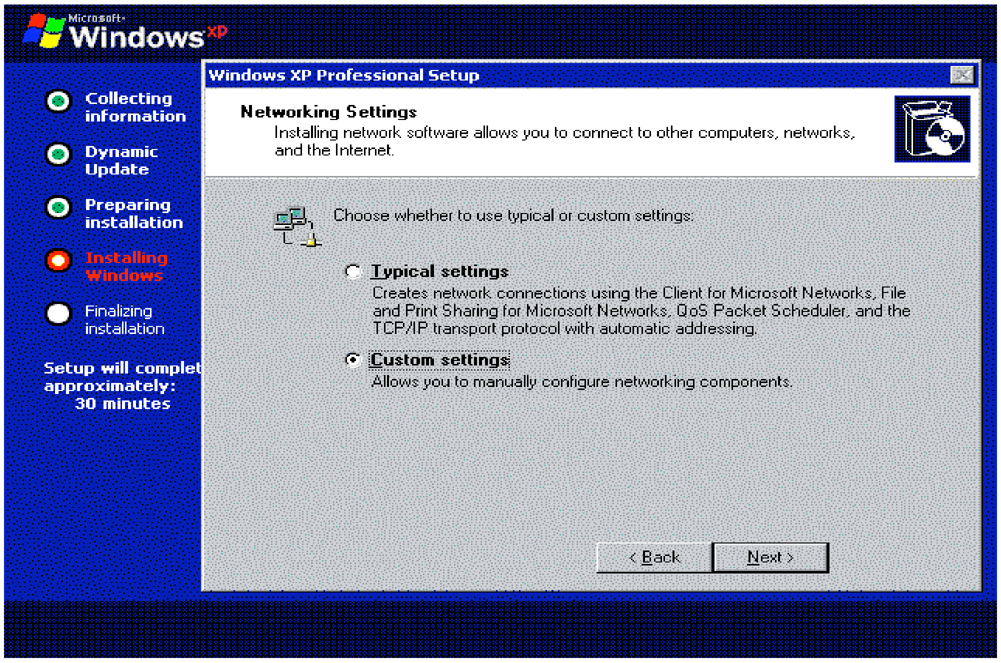
The wizard is located in the System Tools folder. For a command-line version that system administrators can use on multiple computers at once, try the User State Migration Tool (USMT) in the ValueAdd folder.

Customizing Network Settings

Now you need to set up your system so that it can be networked with other systems.Following are the steps:

1. First, you must choose the type of settings you are going to use. Note that it is rarely a good idea to

use **Typical settings** when configuring an application or operating system. It is always a good idea to choose **Custom settings**, as shown in the following screen. When you use this option, you only install options you need. You won't end up installing something you aren't aware of because you chose an option that automatically does this. After you select the **Custom settings** option, click**Next**.



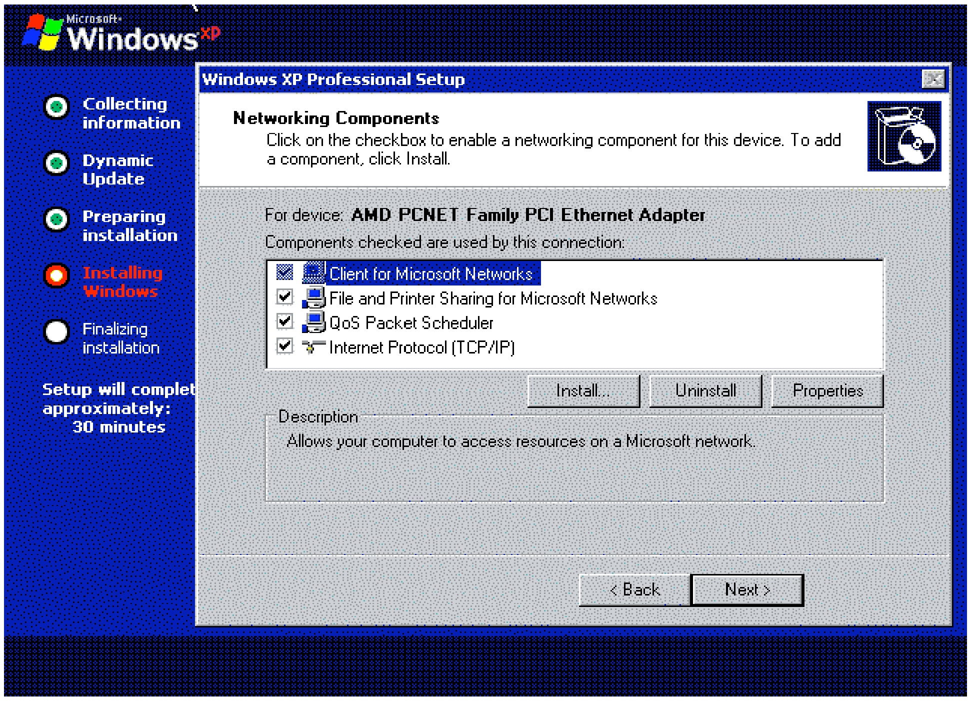
Networking Settings
Installing network software allows you to connect to other computers, networks, and the Internet.

Choose whether to use typical or custom settings:

Typical settings
Creates network connections using the Client for Microsoft Networks, File and Print Sharing for Microsoft Networks, QoS Packet Scheduler, and the TCP/IP transport protocol with automatic addressing.

[✓ Custom settings]
Allows you to manually configure networking components.

2.Windows no longer tries to install IPX/SPX, so there is nothing in the custom settings that you need to remove. This is a great time to setup yourlocal IP address if you are not using a DHCP server in your environment. The assumption here is that you are not going to plug this test machine into a production environment, so it's safe to add your own IP address. Highlight **Internet Protocol (TCP/IP)** and click the **Properties** tab.



Windows XP Professional Setup

Networking Components
Click on the checkbox to enable a networking component for this device. To add a component, click Install.

For device: **AMD PCNET Family PCI Ethernet Adapter**

Components checked are used by this connection:

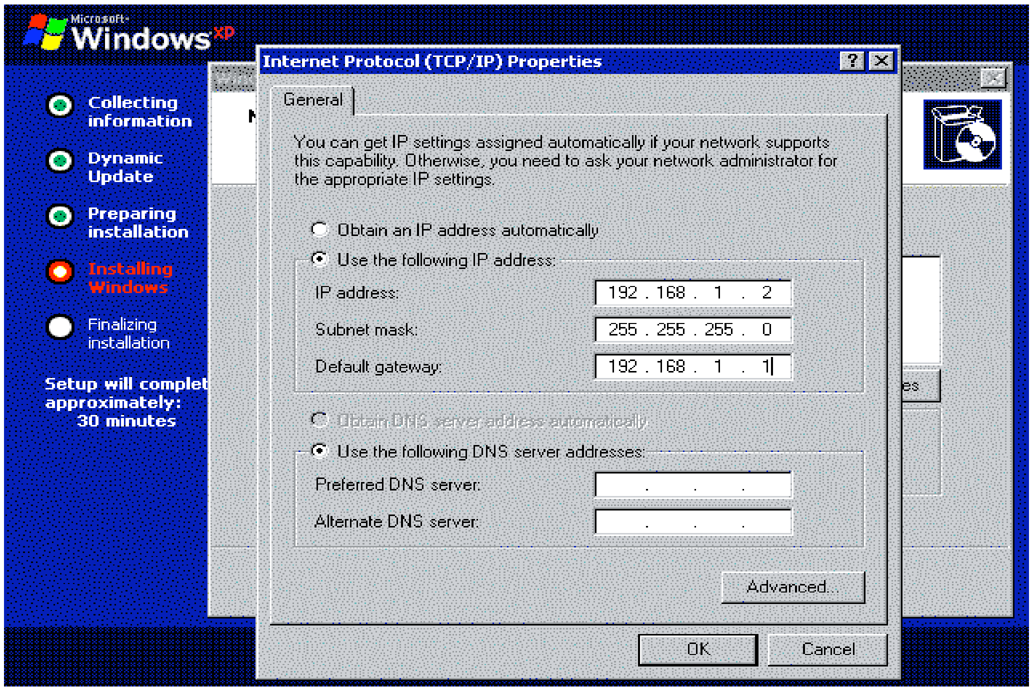
- [✓] Client for Microsoft Networks
- [✓] File and Printer Sharing for Microsoft Networks
- [✓] QoS Packet Scheduler
- [✓] Internet Protocol (TCP/IP)

Install... Uninstall Properties

Description
Allows your computer to access resources on a Microsoft network.

< Back Next >

3. The following screen appears, which allows you to enter your own IP address. For the purposes of this exercise, use a non-routable IP address. Select ‘Use the following IP address’. Enter **192.168.1.2** in the correct fields. Then, enter a standard 24-bit subnet mask of **255.255.255.0**. To make the entire section complete, enter a default gateway setting of **192.168.1.1**. Enter the appropriate DNS server IP addresses for your environment into the fields shown in the following paragraph. You can leave your DNS sever fields blank for this system. Click OK. Click Next.



Internet Protocol (TCP/IP) Properties

General

You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

Obtain an IP address automatically [✓ Use the following IP address:]

IP address:	192 . 168 . 1 . 2
Subnet mask:	255 . 255 . 255 . 0
Default gateway:	192 . 168 . 1 . 1

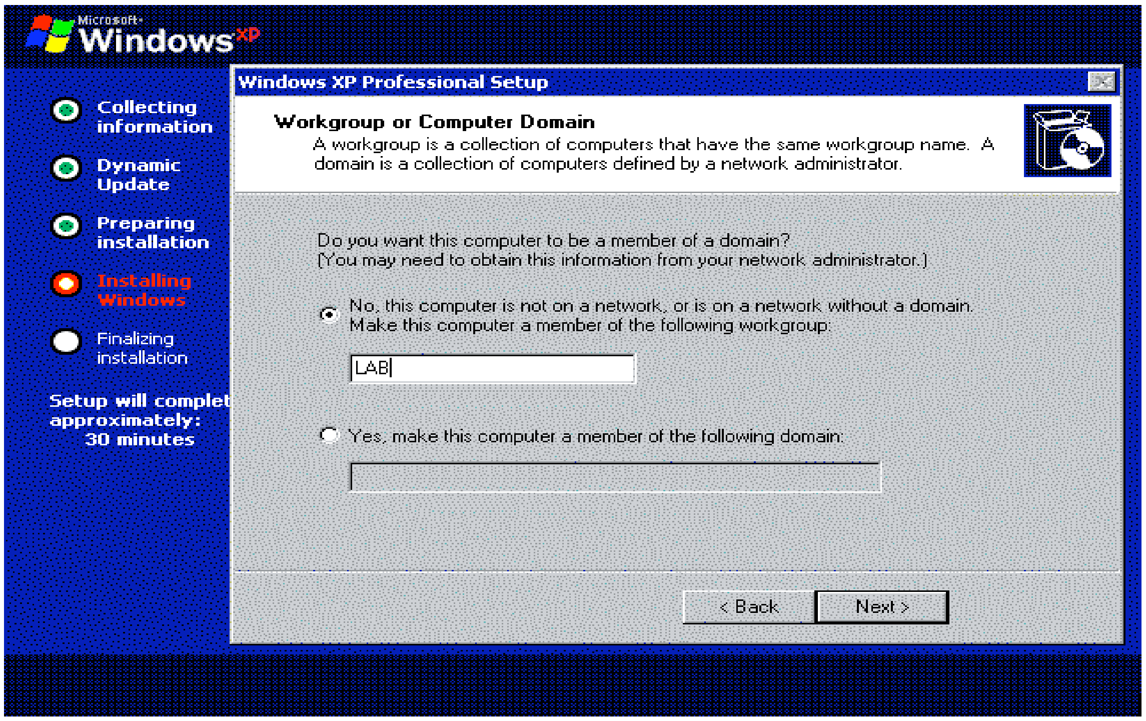
Obtain DNS server address automatically [✓ Use the following DNS server addresses:]

Preferred DNS server:	. . .
Alternate DNS server:	. . .

Advanced...

OK Cancel

4. As previously stated, you are not joining a network or a domain, so just enter a name of your choice and leave the first **No** option enabled (see theillustration that follows). After you have the information entered, click **Next**.



Workgroup or Computer Domain

A workgroup is a collection of computers that have the same workgroup name. A domain is a collection of computers defined by a network administrator.

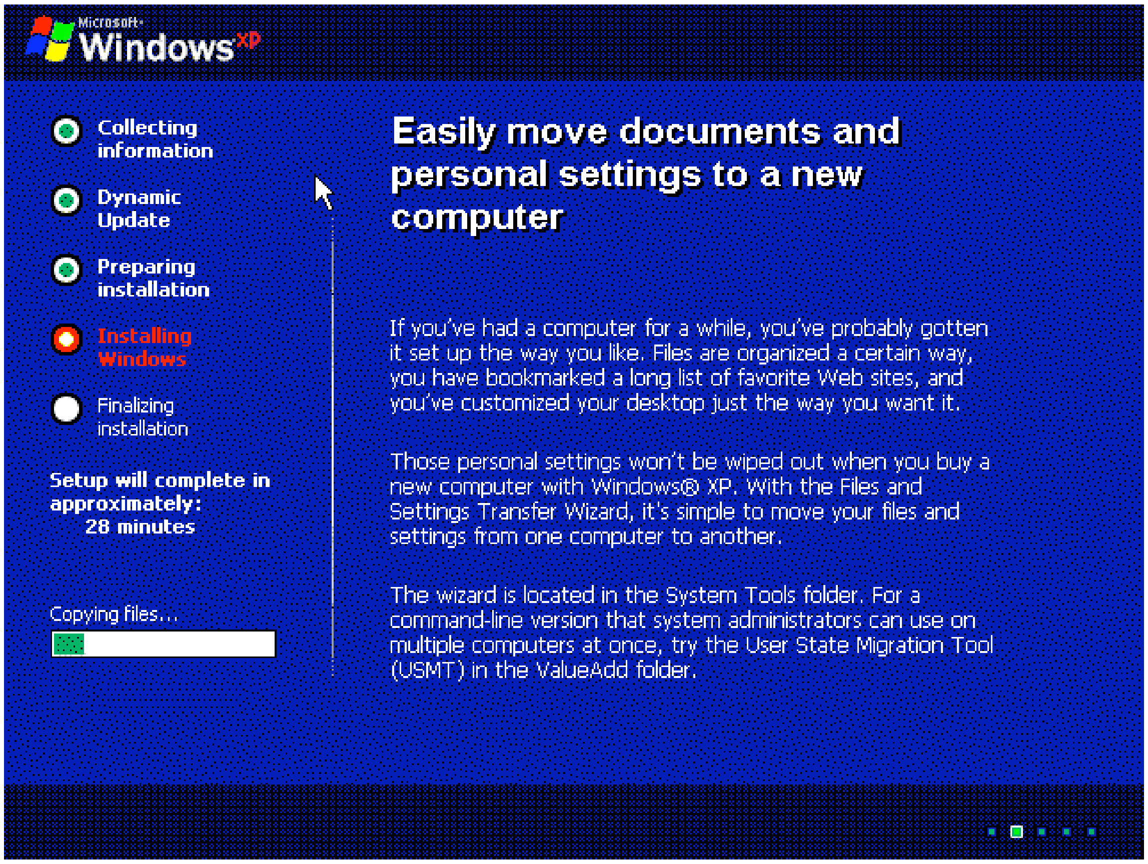
Do you want this computer to be a member of a domain?
(You may need to obtain this information from your network administrator.)

[✓ No, this computer is not on a network, or is on a network without a domain. Make this computer a member of the following workgroup:]

LAB

Yes, make this computer a member of the following domain:

5.Windows completes the networking portion of the installation and moveson to its final tasks. This step takes a long time, so take the opportunity tograb another caffeine-laced beverage.



Setup will complete in approximately:
28 minutes

Copying files...

Easily move documents and personal settings to a new computer

If you've had a computer for a while, you've probably gotten it set up the way you like. Files are organized a certain way, you have bookmarked a long list of favorite Web sites, and you've customized your desktop just the way you want it.

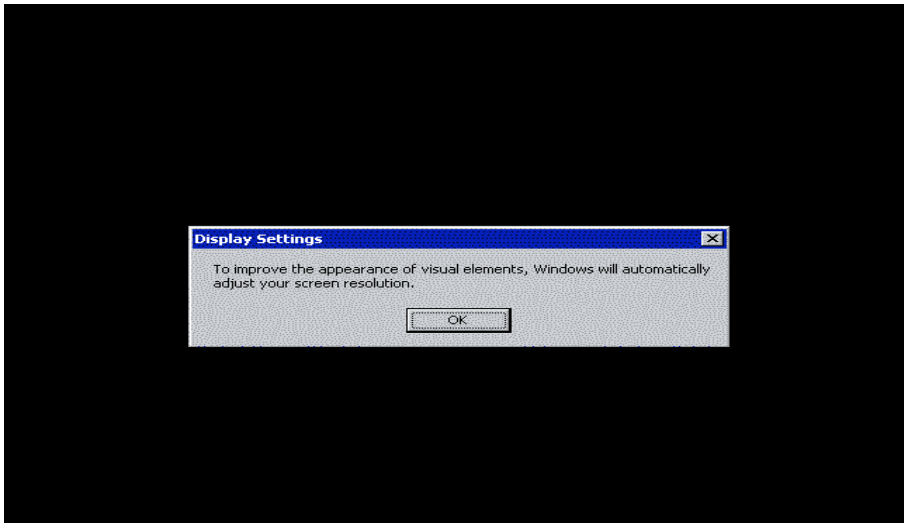
Those personal settings won't be wiped out when you buy a new computer with Windows® XP. With the Files and Settings Transfer Wizard, it's simple to move your files and settings from one computer to another.

The wizard is located in the System Tools folder. For a command-line version that system administrators can use on multiple computers at once, try the User State Migration Tool (USMT) in the ValueAdd folder.

6. If you get the following screen, shout for joy. Congratulations, you havesuccessfully installed Windows XP. Click **Finish**, and then remove the Windows CD-ROM before the system reboots so that you don't accidentally start the install process again. If you accidentally leave the CD-ROM in, and the install process starts again, simply remove the CD-ROM and hard-boot the machine (restart it).

Microsoft®
Windows® xp
Professional
Copyright © 1985-2001
Microsoft Corporation
Microsoft

7. After the next screen comes up, click the **OK** button.

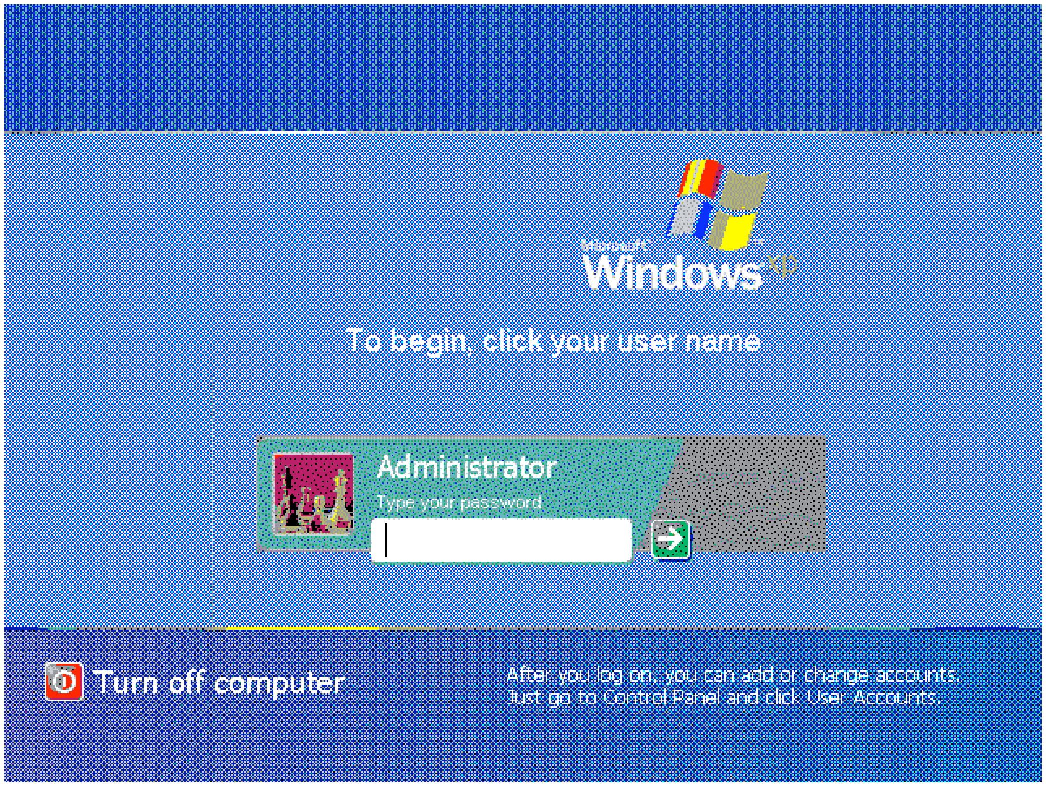


Display Settings

To improve the appearance of visual elements, Windows will automatically adjust your screen resolution.

8. As shown in the next screen, you now need to log in using the Administratoraccount and the password you entered earlier during the install. After you have entered the appropriate credentials, click **OK**.

Note: Depending on your version of the software you might get several screens aboutconnecting to the Internet and registering Microsoft before you get the login screen.



After you log on, you can add or change accounts. Just go to Control Panel and click User Accounts.

Note: At this stage, the base installation of Windows XP is installed. The instructions that followshow you how to upgrade to Service Pack 2, which is recommended for this class.

Installing Service Pack 2

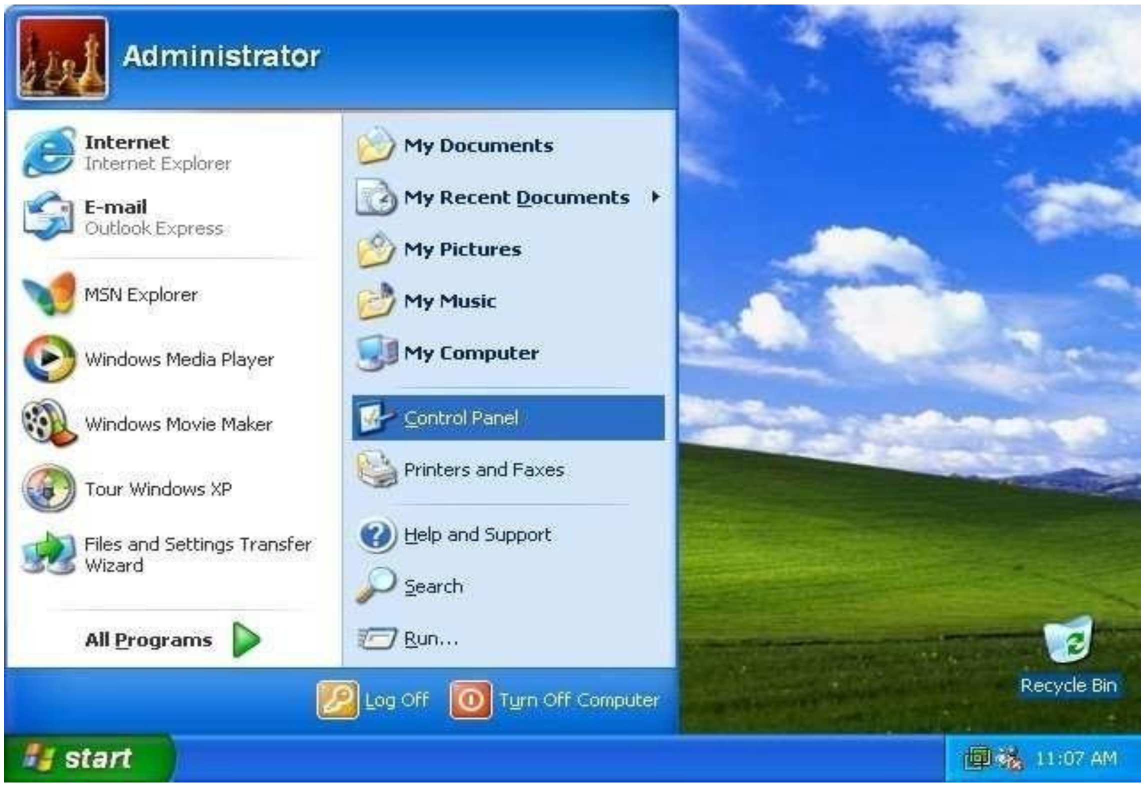
If you do not have SP2 on a CD, you need to get Internet access setup, so that you can patch thisbox to SP2. SP2 provides several functional patches, which is why you want to upgrade to it.

The first thing you need to do is verify that your NIC (Network Interface Card) is working and that you have connectivity. Ensure that your system has the NIC plugged into a switch or hubthat is connected to the Internet. If you have a DHCP (Dynamic Host Configuration Protocol) server on your network, you should be able to automatically pull an IP address. Otherwise, you need to statically assign the appropriate IP address for your system.

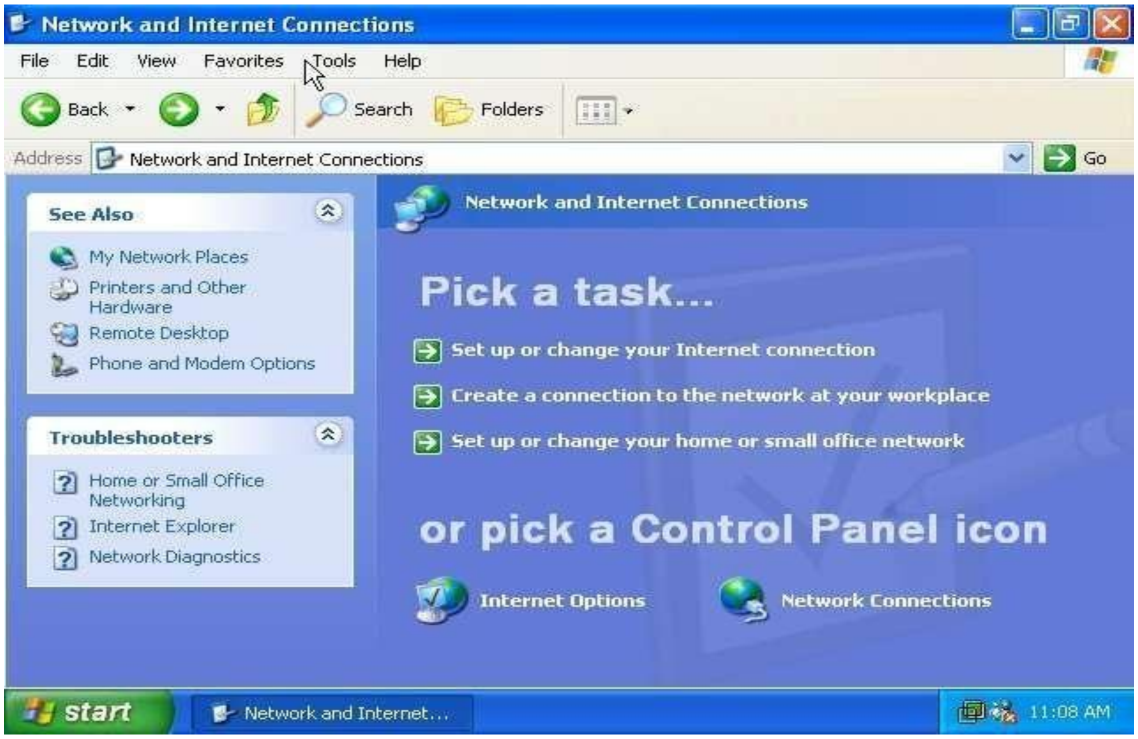
Note: You will need to change the address scheme you entered earlier if you need toconnect to the Internet to download SP2.

Following are the steps for upgrading to SP2:

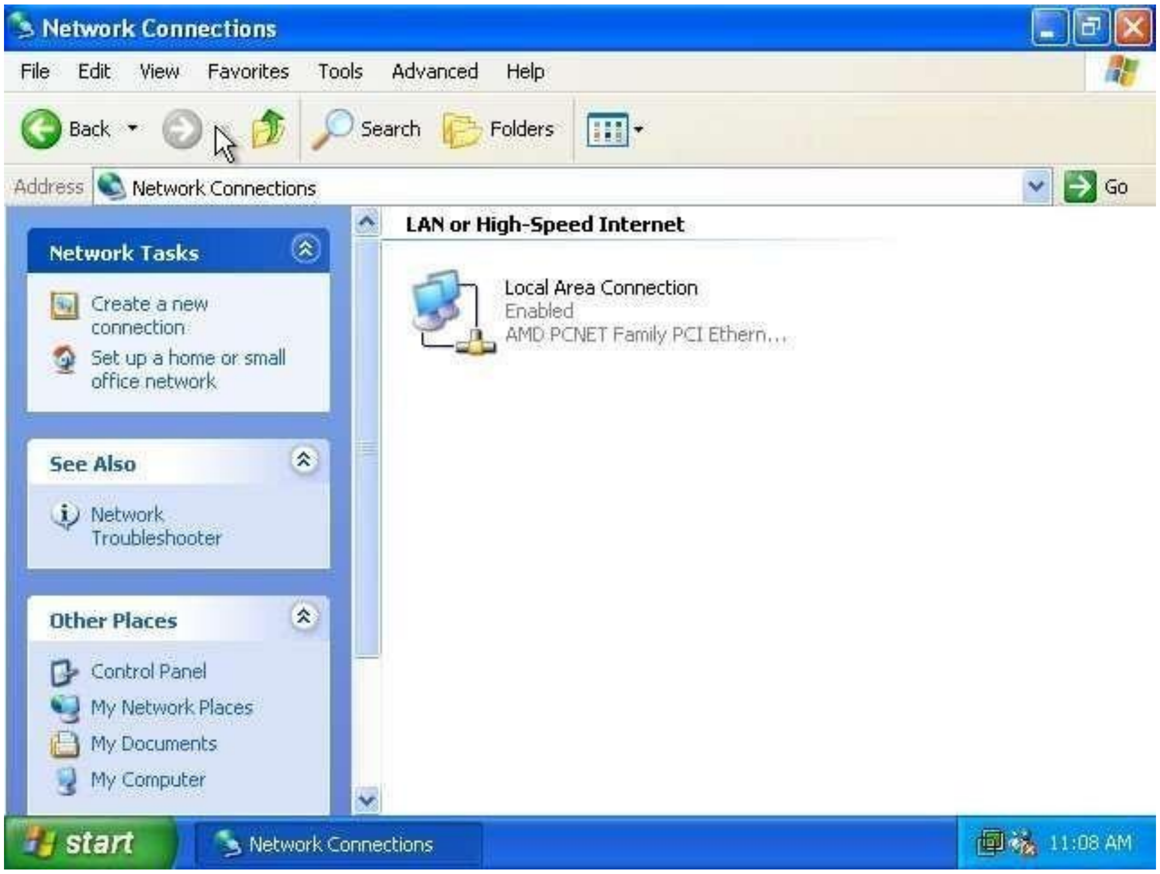
1. Left click on the Start button located in the lower left portion of yourscreen. Then highlight and left click **Control Panel**.



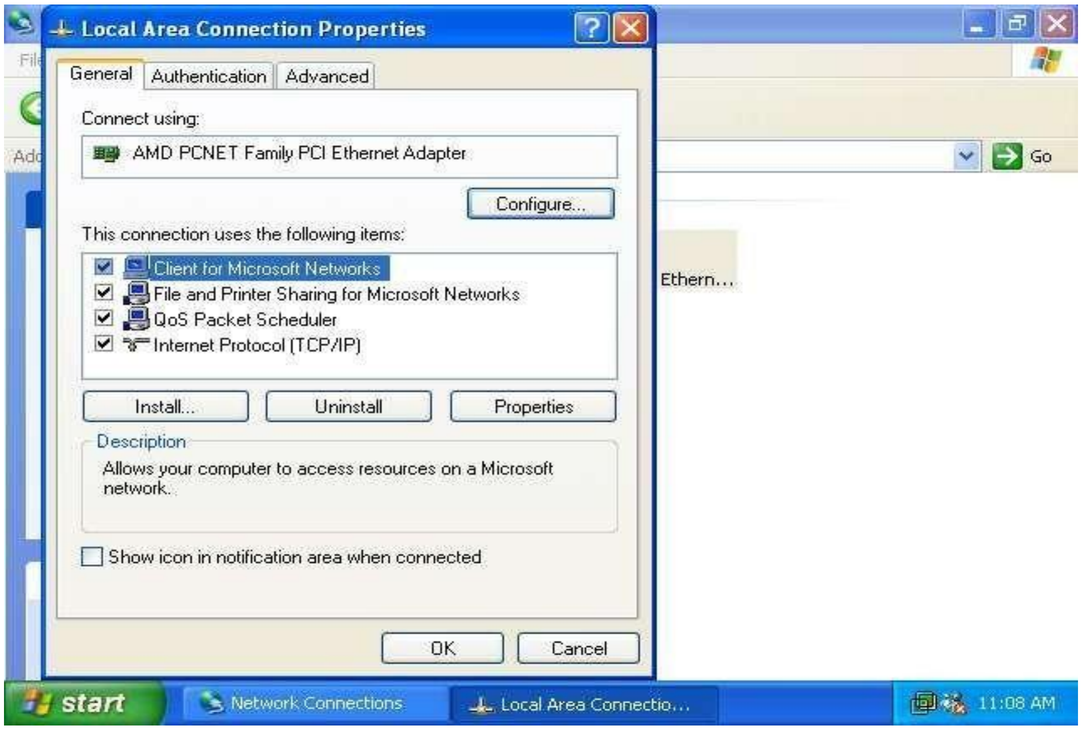
2. Now click on **Network and Internet Connections.** The following screenshould appear. Left click on **Network Connections**.


Network and Internet Connections
File Edit View Favorites Tools Help
Back Search Folders
Address Network and Internet Connections Go
See Also
My Network Places
Printers and Other Hardware
Remote Desktop
Phone and Modem Options
Troubleshooters
Home or Small Office Networking
Internet Explorer
Network Diagnostics
Network and Internet Connections
Pick a task...
Set up or change your Internet connection
Create a connection to the network at your workplace
Set up or change your home or small office network
or pick a Control Panel icon
Internet Options
Network Connections
start
Network and Internet...
11:08 AM

3. The following screen should appear. If you do not see a Local Area Connection, you do not have a NIC (Network Interface Card) installed orproperly working. If this is the case you will need to check with your NICvendor's documentation on getting your particular card installed in Windows XP.Most modern NICs are fully compatible with Windows XP.

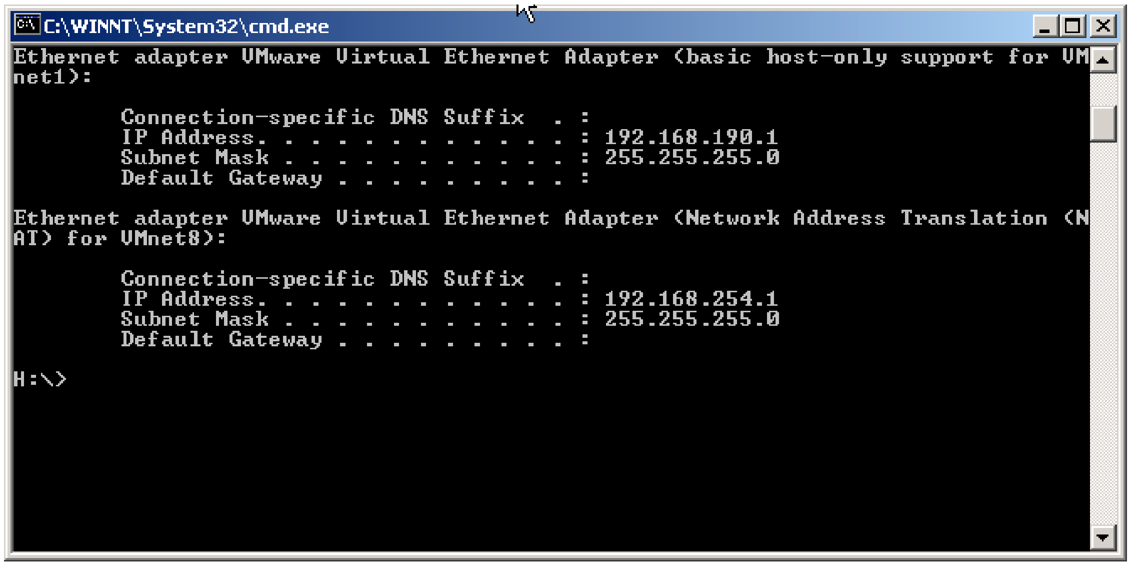


4. Right click on **Local Area Connection** and then left click on **Properties**in the menu that appears. This screen shows you the different configuration items that this particular interface uses. To exit this, click on **OK**.


Local Area Connection Properties
General
Authentication
Advanced
Connect using:
AMD PCNET Family PCI Ethernet Adapter
Configure...
This connection uses the following items:
Client for Microsoft Networks
File and Printer Sharing for Microsoft Networks
QoS Packet Scheduler
Internet Protocol (TCP/IP)
Install...
Uninstall
Properties
Description
Allows your computer to access resources on a Microsoft network.
Show icon in notification area when connected
OK
Cancel
Go
Ethern...
start
Network Connections
Local Area Connectio...
11:08 AM

5. After the Hardware Manager is properly setup, you need to validate that the IP address we initially configured is on your system. Click on **Start, Run,** and thentype **cmd**. Type **ipconfig**. If you see an IP address next to the NIC, you can proceed.

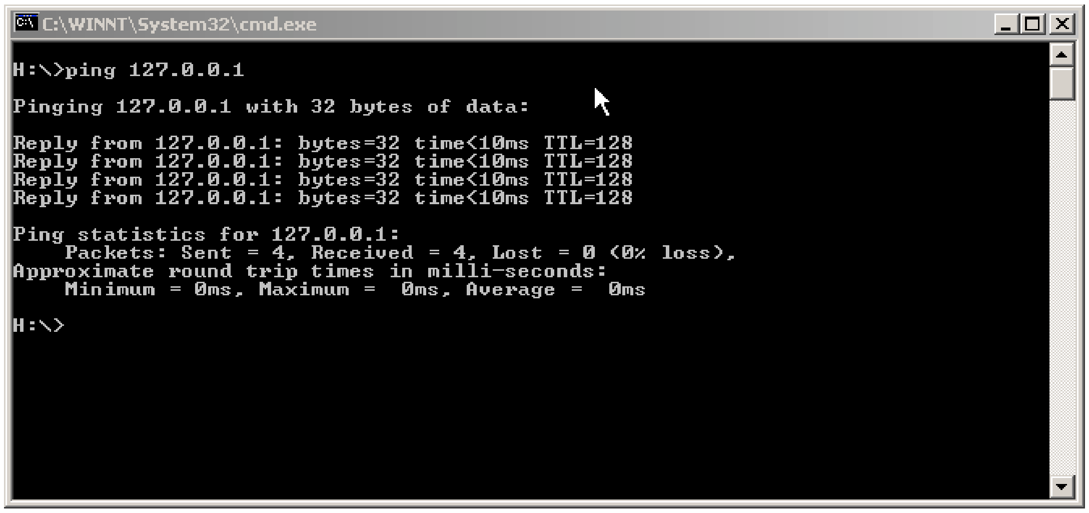
If you do not see an IP address, or you see the address with **169.254.30.x**, you didn't pull an IP address from your DHCP server or the IP configuration step we preformed earlier was not successful. You will need to manually add an IP address by repeating thesteps described during the installation of Windows XP. If you need to repeat these stepsto add an appropriate IP address for your network, do so now.



C:\WINNT\System32\cmd.exe

```
Ethernet adapter VMware Virtual Ethernet Adapter (basic host-only support for VM
net1):

        Connection-specific DNS Suffix  . :
        IP Address. . . . . . . . . . . . : 192.168.190.1
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . :

Ethernet adapter VMware Virtual Ethernet Adapter (Network Address Translation (N
AT) for VMnet8):

        Connection-specific DNS Suffix  . :
        IP Address. . . . . . . . . . . . : 192.168.254.1
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . :

H:\>
```

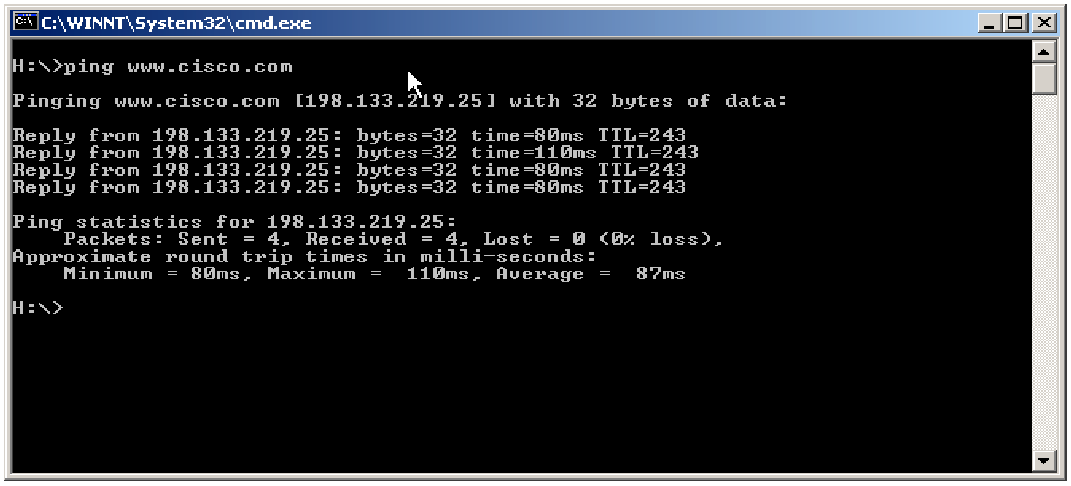
6. Next, you need to verify connectivity to the Internet. To make sure yourlocal IP stack is functioning correctly, you can PING the loopback adapter. To do this, open another command window by selecting **Start,Run**, and then type **cmd**. Then, type **ping 127.0.0.1**, as shown in the following screen.



```
H:\>ping 127.0.0.1

Pinging 127.0.0.1 with 32 bytes of data:

Reply from 127.0.0.1: bytes=32 time<10ms TTL=128
Reply from 127.0.0.1: bytes=32 time<10ms TTL=128
Reply from 127.0.0.1: bytes=32 time<10ms TTL=128
Reply from 127.0.0.1: bytes=32 time<10ms TTL=128

Ping statistics for 127.0.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum =  0ms, Average =  0ms

H:\>
```

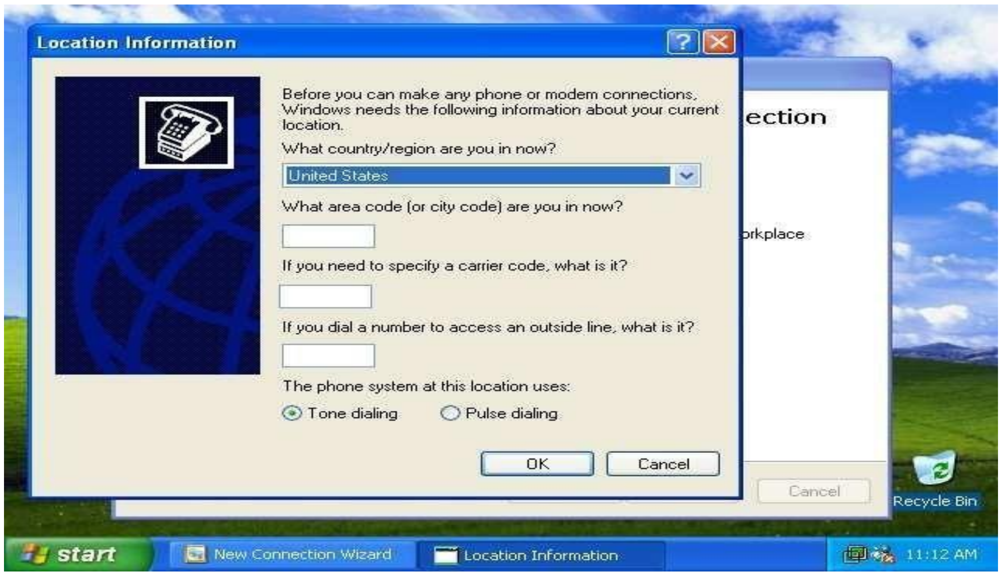
7. .As you can see, there is connectivity to the local IP stack. This shows thatthe TCP/IP stack is functioning correctly. To verify Internet connectivity and that the DNS settings are working correctly, ping a web site. The IP address used in the following screen is not valid. You need to ping a validIP address. For example, pinging **www.sans.org** should work.

Note: If you are not on a network that is connected to the Internet this step will not work. Also, if you are properly connected to an Internet accessible network and you used the IP address we supplied, and it does not match the network information of your network, this step will not work. If the later is thecase, please change your IP address to match the information that is appropriate for your environment.



```
H:\>ping www.cisco.com

Pinging www.cisco.com [198.133.219.25] with 32 bytes of data:

Reply from 198.133.219.25: bytes=32 time=80ms TTL=243
Reply from 198.133.219.25: bytes=32 time=110ms TTL=243
Reply from 198.133.219.25: bytes=32 time=80ms TTL=243
Reply from 198.133.219.25: bytes=32 time=80ms TTL=243

Ping statistics for 198.133.219.25:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 80ms, Maximum =  110ms, Average =  87ms

H:\>
```

8. If you get an **Unable to resolve name** message, you need to validate thatyou have entered your DNS servers into the TCP/IP properties of your NIC correctly.


Location Information

Before you can make any phone or modem connections, Windows needs the following information about your current location.

What country/region are you in now? United States

What area code (or city code) are you in now?

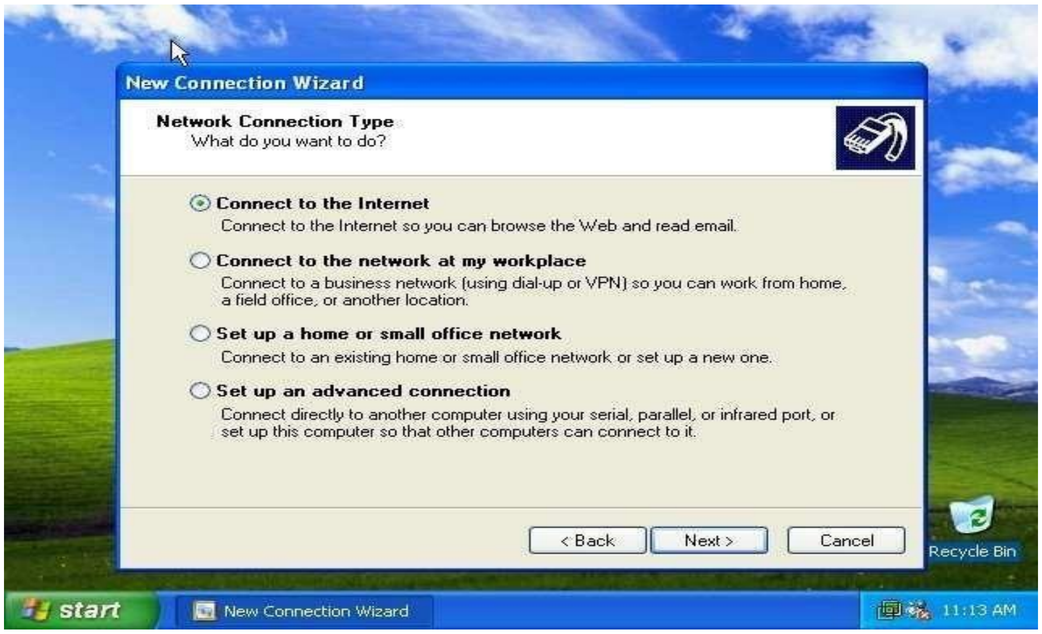
If you need to specify a carrier code, what is it?

If you dial a number to access an outside line, what is it?

The phone system at this location uses: [✓ Tone dialing] Pulse dialing

9. To get your browser functioning, double-click the **Internet Explorer** iconon your desktop and follow the wizard's instructions. In the first window click on **Cancel** since we will not be using a modem.

10. In the next window choose the appropriate option for your home environment. If you are part of a LAN (Local Area Network), choose thefirst option (**Connect to the Internet)**. Then, click **Next**.


New Connection Wizard

Network Connection Type
What do you want to do?

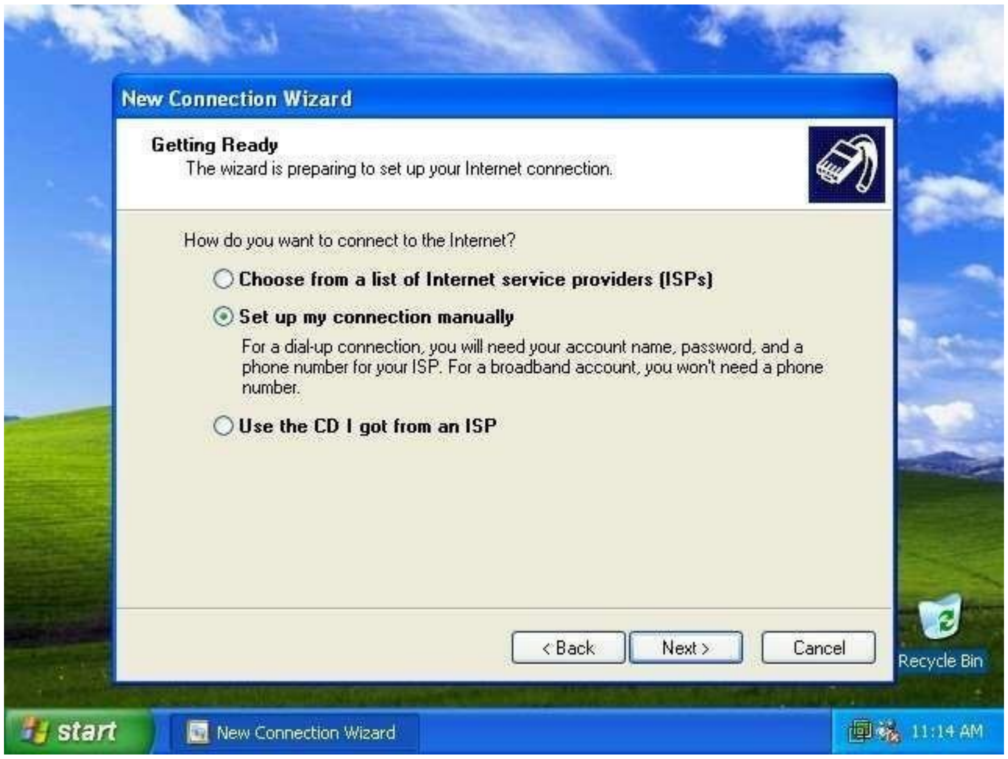
[✓ Connect to the Internet]
Connect to the Internet so you can browse the Web and read email.

Connect to the network at my workplace
Connect to a business network (using dial-up or VPN) so you can work from home, a field office, or another location.

Set up a home or small office network
Connect to an existing home or small office network or set up a new one.

Set up an advanced connection
Connect directly to another computer using your serial, parallel, or infrared port, or set up this computer so that other computers can connect to it.

11. In the next screen, choose **Set up my connection manually**. Click **Next**.


New Connection Wizard

Getting Ready

The wizard is preparing to set up your Internet connection.

How do you want to connect to the Internet?

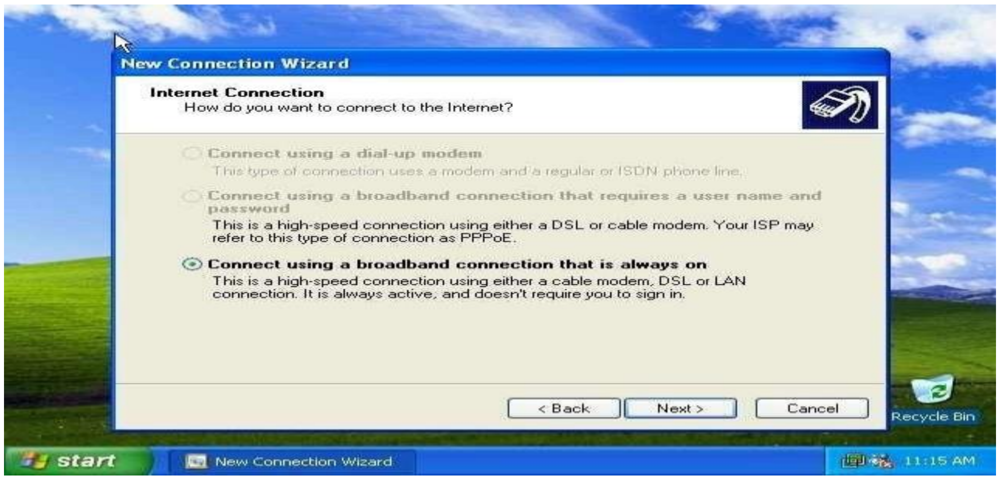
( ) Choose from a list of Internet service providers (ISPs)

[✓ Set up my connection manually]

For a dial-up connection, you will need your account name, password, and a phone number for your ISP. For a broadband account, you won't need a phone number.

( ) Use the CD I got from an ISP

12. Do not change any options for the next screen. Leave it alone, and simplyclick **Next**.


New Connection Wizard

Internet Connection

How do you want to connect to the Internet?

( ) Connect using a dial-up modem

This type of connection uses a modem and a regular or ISDN phone line.

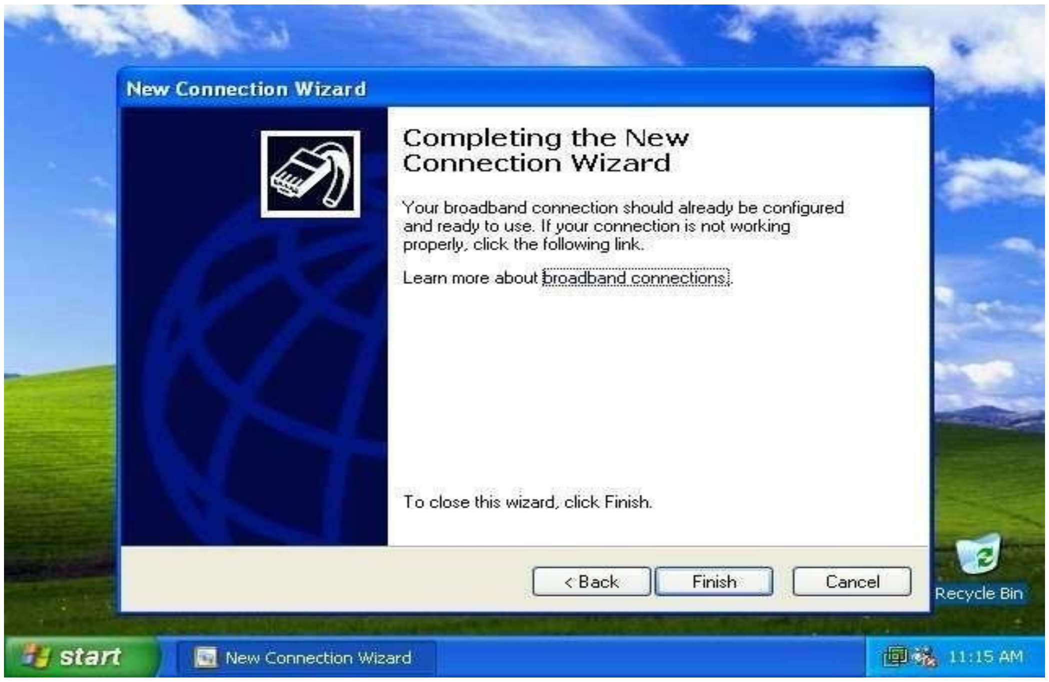
( ) Connect using a broadband connection that requires a user name and password

This is a high-speed connection using either a DSL or cable modem. Your ISP may refer to this type of connection as PPPoE.

[✓ Connect using a broadband connection that is always on]

This is a high-speed connection using either a cable modem, DSL or LAN connection. It is always active, and doesn't require you to sign in.

13. You have now setup Internet Explorer for web surfing, so click **Finish.** You can now go to Microsoft's web site to download and install ServicePack 2.


New Connection Wizard

Completing the New Connection Wizard

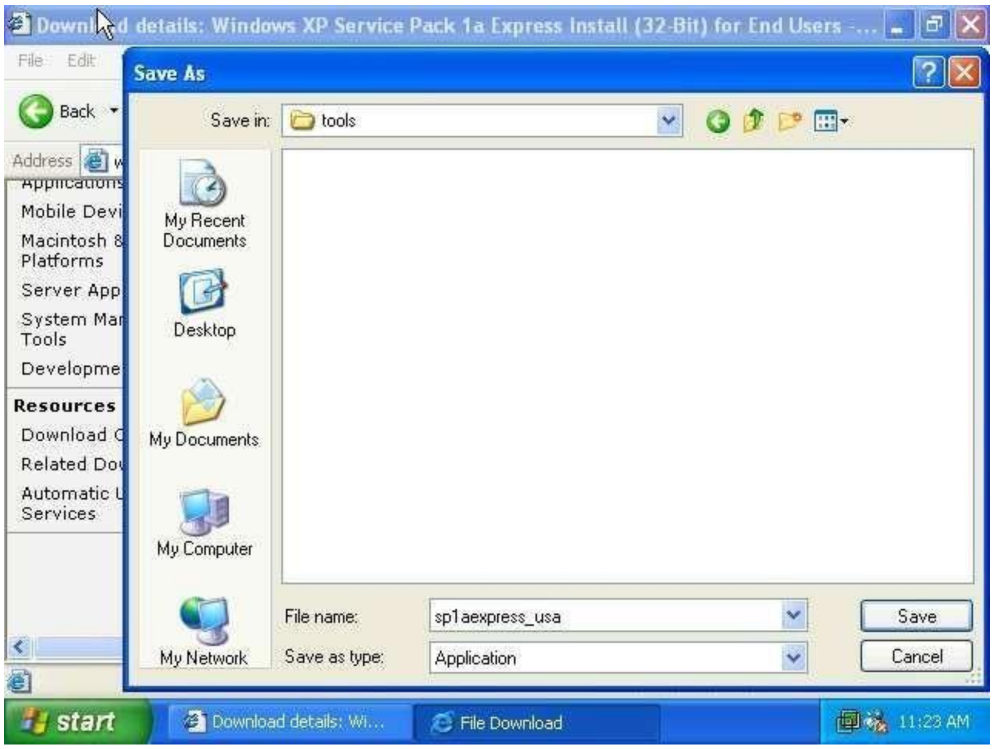
Your broadband connection should already be configured and ready to use. If your connection is not working properly, click the following link.

Learn more about broadband connections.

To close this wizard, click Finish.

14. The last thing we need to do prior to installing anything on our system, including the Service Pack, is to create a folder that we will be storing allof the installation executables throughout this book.

Click on **Start** then **My Computer**. Double click on **C**. Now right click anywhere in the window and left click **New** then **Folder**. Namethe new folder **tools** as shown below


Download details: Windows XP Service Pack 1a Express Install (32-Bit) for End Users -...

Save As

Save in: tools

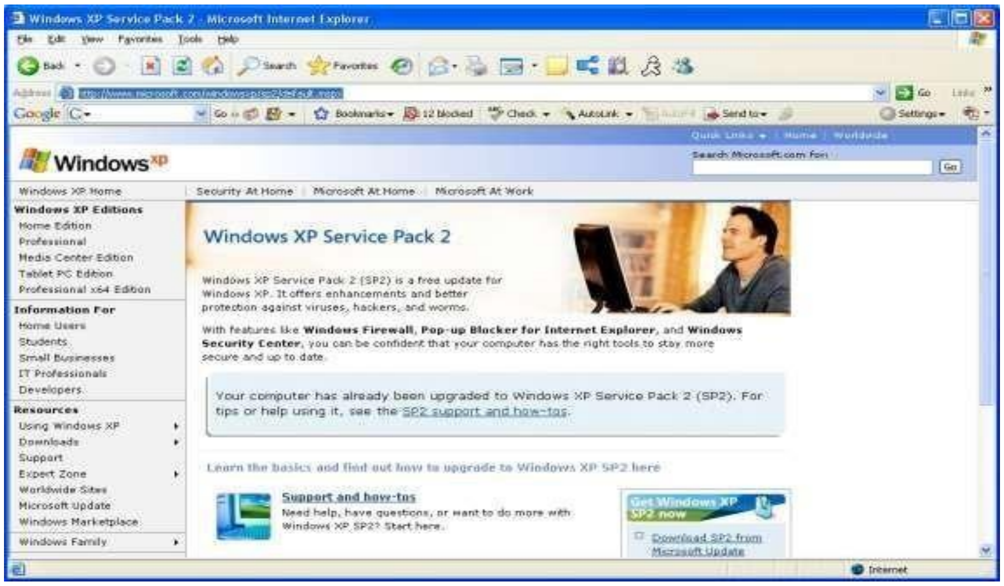
File name: sp1aexpress_usa

Save as type: Application

15 .To install Service pack 2, go to the following URL:

http://www.microsoft.com/windowsxp/sp2/default.mspx

The Microsoft web site is shown in the following screen.



Windows XP Service Pack 2

Windows XP Service Pack 2 (SP2) is a free update for Windows XP. It offers enhancements and better protection against viruses, hackers, and worms.

With features like **Windows Firewall, Pop-up Blocker for Internet Explorer**, and **Windows Security Center**, you can be confident that your computer has the right tools to stay more secure and up to date.

Your computer has already been upgraded to Windows XP Service Pack 2 (SP2). For tips or help using it, see the SP2 support and how-tos.

Learn the basics and find out how to upgrade to Windows XP SP2 here

Support and how-tos
Need help, have questions, or want to do more with Windows XP SP2? Start here.

16. Ensure that **English** is the Selected Language and click **Go.**

17. You can now select either **Express** or **Network Installations**. Both options work, but you should choose the **Network Installation** option.Next, you should download Service Pack 2 into the **tools** directory youcreated earlier or a different directory that you create.

18. After the download is completed, double-click the SP2 executable andfollow the given prompts for installation.

19. Read the License Agreement carefully. When you understand and agreewith it, click on the button next to **I Agree** then click **Next.**

20. Because you do not need to uninstall this Service Pack, you can check the **Do Not Archive Files…** option.

21. Now you get to wait and watch. It is important to note that this process cantake an extremely long time, even if it seems your system has locked up, itmost likely has not. After the Service Pack is completely installed, reboot your machine and you are ready to move on to the next section.

You have now successfully completed the installation of Windows

RESULT:

Thus the installation of Windows is done successfully.

Ex.No:2.a **Date:**	**BASICS OF UNIX COMMANDS**
	INTRODUCTION TO UNIX

AIM:
To study about the basics of UNIX

UNIX:
It is a multi-user operating system. Developed at AT & T Bell Industries, USA in 1969.

Ken Thomson along with Dennis Ritchie developed it from MULTICS (MultiplexedInformation and Computing Service) OS.
By1980, UNIX had been completely rewritten using C language.

LINUX:
It is similar to UNIX, which is created by Linus Torualds. All UNIX commands worksin Linux. Linux is a open source software. The main feature of Linux is coexisting with
other OS such as windows and UNIX.

STRUCTURE OFA LINUXSYSTEM:
It consists of three parts.

a. UNIX kernel
b. Shells
c. Tools and Applications

UNIX KERNEL:
Kernel is the core of the UNIX OS. It controls all tasks, schedule all Processesand carries out all the functions of OS.

Decides when one programs tops and another starts.

SHELL:
Shell is the command interpreter in the UNIX OS. It accepts command from theuser and analyses and interprets them

Ex.No:2.b	**BASICS OF UNIX COMMANDS**
Date:	**BASIC UNIX COMMANDS**

AIM:
To study of Basic UNIX Commands and various UNIX editors such as vi, ed,ex and EMACS.

CONTENT:
Note: Syn->Syntax

a) date
–used to check the date andtime Syn:$date

Format	Purpose	Example	Result
+%m	To display only month	$date+%m	06
+%h	To display month name	$date+%h	June
+%d	To display day of month	$date+%d	O1
+%y	To display last two digits of years	$date+%y	09
+%H	To display hours	$date+%H	10
+%M	To display minutes	$date+%M	45
+%S	To display seconds	$date+%S	55

b) cal
–used to display the calendar Syn:$cal 2 2009

c) echo
–used to print the message on the screen. Syn:$echo "text"

d) l ls
–used to list the files. Your files are kept in directory.

Syn:$lsls–s
All files (include files with prefix)
ls–l Lodetai (provide file statistics)
ls–t Order by creation time
ls– u Sort by access time (or show when last accessed together with –l)
ls–s Order by sizels–
r Reverse order
ls–f Mark directories with /,executable with* , symbolic links with @, local sockets with =, named pipes(FIFOs)withls–s Show file size
ls– h" Human Readable", show file size in Kilo Bytes & Mega Bytes (h can be used together with –l or)

ls[a-m]*List all the files whose name begin with alphabets From „a' to „m'
ls[a]*List all the files whose name begins with „a' or „A'
Eg:$ls>my list Output of „ls' command is stored to disk file named „my list'

e) lp
–used to take printoutsSyn:$lp filename
f) man
–used to provide manual help on every UNIX commands.
Syn:$manunix command
$man cat

g) who & **whoami**
–it displays data about all users who have logged into the system currently. The next commanddisplays about current user only.
Syn:$who$whoami

h) uptime
–tells you how long the computer has been running since its last reboot or power-off.
Syn:$uptime

i) uname
–it displays the system information such as hardware platform, system name and processor, OStype.
Syn:$uname–a

j) hostname
–displays and set system hostname Syn:$ hostname

k) bc
–stands for „best calculator'

```
$bc                  $ bc           $ bc              $ bc
10/2*3                              ibase=2
15                   Scale=1        obase=16          sqrt(196)

3.35                                11010011          14 quit
Quit                                89275
                                     1010
                                    Ā
                                    Quit


$bc                            $ bc-l scale=2
for(i=1;i<3;i=i+1)I

1                               s(3.14)
2                               0
3 quit
```

FILE MANIPULATION COMMANDS

a) **cat–**this create, view and concatenatefiles. **Creation**:
Syn:$cat>filename

Viewing:
Syn:$cat filename
Add text to an existing file:
Syn:$cat>>filename

Concatenate:
Syn:$catfile1file2>file3
$catfile1file2>>file3 (no over writing of file3)

b) **grep–**used to search a particular word or pattern related to that word from the file.
Syn:$grep search word
filename Eg:$grep anu student

c) **rm–**deletes a file from the filesystem Syn:$rm filename

d) **touch–**used to create a blank file.
Syn:$touch file names

e) **cp–**copies the files or directoriesSyn:$cpsource file destination
file Eg:$cp student stud

f) **mv–**to rename the file or directorysyn:$mv old file new file
Eg:$mv–i student student list(-i prompt when overwrite)

g) **cut–**it cuts or pickup a given number of character or fields of the file.
Syn:$cut<option><filename>
Eg: $cut –c filename
$cut–c1-10emp
$cut–f 3,6emp
$ cut –f 3-6 emp
-c cutting columns
-f cutting fields

h) **head–**displays10 lines from the head(top)of a given fileSyn:$head filename
Eg:$head student
To display the top two lines:

Syn:$head-2student

i) **tail–**displays last 10 lines of the fileSyn:$tail filename
Eg:$tail student
To display the bottom twolines; Syn:$tail -2 student

j) **chmod–**used to change the permissions of a file or directory.Syn:$ch mod category operation permission file Where, Category–is the user type
Operation–is used to assign or remove permissionPermission–is the type of permission
File–are used to assign or remove permission all

Examples:

$chmodu-wx student
Removes write and execute permission for users
$ch modu+rw,g+rwstudent
Assigns read and write permission for users and groups
$chmodg=rwx student
Assigns absolute permission for groups of all read, write and execute permissions

k) **wc–**it counts the number of lines, words, character in a specifiedfile(s) with the options as –l,-w,-c

Category	Operation	Permission
u– users g–group o– others	+assign -remove =assign absolutely	r– read w–write x-execute

Syn: $wc –l filename
$wc–w filename
$wc–c filename

Ex.No:2.c **Date:**	**BASICS OF UNIX COMMANDS**
	UNIX EDITORS

AIM:
To study of various UNIX editors such as vi, ed, ex and EMACS.

CONCEPT:
Editor is a program that allows user to see a portions a file on the screen and modifycharacters and lines by simply typing at the current position. UNIX supports variety of Editors.They are:
ed ex vi EMACS
Vi- vi is stands for "visual".vi is the most important and powerful editor.vi is a full screen editorthat allows user to view and edit entire document at the same time.vi editor was written in the University of California, at Berkley by Bill Joy, who is one of the co-founder of Sun Microsystems.

Features of vi:
It is easy to learn and has more powerful
features. Itworksgreatspeedandiscasesensitive.vihaspowerfulundofunctionsandhas3modes:

1. Command mode
2. Insert mode
3. Escape or ex mode

In command mode, no text is displayed on the screen.
In Insert mode, it permits user to edit insert or replace text.
In escape mode, it displays commands at commandline.
Moving the cursor with the help of h, l, k, j, I, etc

EMACS Editor
Motion Commands:
M-> Move to end of file
M-< Move to beginning of file
C-v Move forward a screen M –v
Move backward a screen C –n Move tonext line C-p Move to previous line
Move to the beginning of the lineC-e Move to the end of the line
C-f Move forward a character
Move backward a characterM-f Move forward a word
M-b Move backward a word

Deletion commands:

DEL	delete the previous character C-d
	Delete the current character M-DEL
	Delete the previous word
M-d	delete the next word
C-x DEL	deletes the previous sentence
M-k	delete the rest of the current sentence
C-k	delete the rest of the current line
C-xu	undo the lasted it change

Search and Replace in EMACS:

y Change the occurrence of the pattern

n Don't change the occurrence, but look for the other q Don't change. Leave query replace completely

! Change this occurrence and all others in the file

RESULT:

Thus the programs implementing basics of UNIX commands is executed successfully.

Ex.No:2d **Date:**	**SIMPLE SHELL PROGRAMS**

AIM:

To write simple shell programs by using conditional, branching and looping statements.

1. Write a Shell program to check the given number is even or oddALGORITHM:

SEPT 1: Start the program.
STEP 2: Read the value of n.
STEP 3: Calculate „r=expr$n%2‟.
STEP 4: If the value of r equals 0 then print the number is even.
STEP 5: If the value of r not equal to 0then print the number is odd.

PROGRAM:

```
echo "Enter the Number"
read n
r=$(expr $n % 2)
if [ $r -eq 0 ]; then
   echo "$n is Even number"
else
   echo "$n is Odd number"
fi
```

OUTPUT

```
Enter the Number 2
 2 is Even number
```

2. Write a Shell program to check the given year is leap year or not

ALGORITHM:

SEPT 1: Start the program.
STEP 2: Read the value of year.
STEP 3: Calculate „b=expr $y%4‟.
STEP 4: If the value of b equals 0 then print the year is a leap year
STEP 5: If the value of r not equal to 0 then print the year is not a leap year.

PROGRAM:

```
echo "Enter the year:"
read y
b=$(expr $y % 4)
if [ $b -eq 0 ]; then
   echo "$y is a leap year"
else
   echo "$y is not a leap year"
fi
```

OUTPUT

```
Enter the year 2004
2004 is a leap year
```

3. Write a Shell program to find the factorial of a number.

ALGORITHM:

SEPT 1: Start the program.
STEP 2: Read the value of n.
STEP 3: Calculate „i=expr $n-1".
STEP 4: If the value of i is greater than 1 then calculate „n=expr $n * $i" and „i=expr $i – 1"
STEP 5: Print the factorial of the given number.

PROGRAM:

```
echo "Enter a Number"
read n
i=`expr $n - 1`
p=$n
while [ $i -ge 1 ]
do
   p=`expr $p \* $i`
   i=`expr $i - 1`
done
echo "The Factorial of the given Number is $p"
```

<u>OUTPUT</u>

```
Enter a Number 4
The Factorial of the given Number is 24
```

4. Write a Shell program to swap the two integers

ALGORITHM:

SEPT 1: Start the program.
STEP 2: Read the value of a, b.
STEP 3: Calculate the swapping of two values by using a temporary variable temp.
STEP 4: Print thevalue of a and b.

PROGRAM:

```
echo "Enter Two Numbers:"
read a b
temp=$a
a=$b
b=$temp
echo "After swapping:"
echo $a $b
```

OUTPUT

```
Enter Two Numbers 2 3
after swapping 3 2
```

RESULT:

Thus the programs of simple shell programs by using conditional, branching and looping statements is implemented successfully.

Ex.No:3 **Date:**	Programs using the following system calls of UNIX operating systemfork, exec, getpid, exit, wait, close

AIM:

To write C Programs using the following system calls of UNIX operating system fork, exec,getpid, exit, wait, close.

PROGRAM FOR SYSTEM CALLS OF UNIX OPERATING SYSTEM (fork, getpid, exit)

ALGORITHM:

```
STEP 1: Start the program.
STEP 2: Declare the variables pid,pid1,pid2.
STEP 3: Call fork() system call to create process.
STEP 4: If pid==-1, exit.
STEP 5: Ifpid!=-1 , get the process id using getpid().
STEP 6: Print the process id.
STEP 7:Stop the program
```

PROGRAM:

```
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h> // Added for exit()

int main() {
    int pid, pid1, pid2;
    pid = fork();
    if (pid == -1) {
        printf("ERROR IN PROCESS CREATION \n");
        exit(1);
    }
    if (pid != 0) {
        pid1 = getpid();
        printf("\nThe parent process ID is %d\n", pid1);
    } else {
        pid2 = getpid();
        printf("\nThe child process ID is %d\n", pid2);
    }
    return 0;
}
```

OUTPUT:

```
The parent process ID is 10245
The child process ID is 10249
```

RESULT:

Thus the C Programs using the following system calls of UNIX operating system fork, exec,getpid, exit, wait, close are implemented successfully.

Ex.No:4a Date:	CPU SCHEDULING ALGORITHMS
	PRIORITY

AIM:
To write a C program for implementation of Priority scheduling algorithms.

ALGORITHM:
Step 1: Inside the structure declare the variables.
Step 2: Declare the variable i,j as integer, totwtime and totttime is equal to zero.
Step 3: Get the value of „n' assign p and allocate thememory.
Step 4: Inside the for loop get the value of bursttime and priority.
Step 5: Assign wtime as zero .
Step 6: Check p[i].pri is greater than p[j].pri .
Step 7: Calculate the total of burst time and waiting time and assign as turnaroundtime.
Step 8: Stop the program.

PROGRAM:
```
#include<stdio.h>
#include<stdlib.h>

typedef struct {
    int pno;
    int pri;
    int btime;
    int wtime;
} sp;

int main() {
    int i, j, n;
    int tbm = 0, totwtime = 0, totttime = 0;
    sp *p, t;

    printf("\n PRIORITY SCHEDULING.\n");
    printf("\n enter the no of process:\n");
    scanf("%d", &n);

    p = (sp*)malloc(n * sizeof(sp));

    printf("Enter the burst time and priority:\n");
    for (i = 0; i < n; i++) {
        printf("Process %d: ", i + 1);
        scanf("%d%d", &p[i].btime, &p[i].pri);
        p[i].pno = i + 1;
        p[i].wtime = 0;
    }

    for (i = 0; i < n - 1; i++) {
        for (j = i + 1; j < n; j++) {
            if (p[i].pri > p[j].pri) {
                t = p[i];
                p[i] = p[j];
                p[j] = t;
            }
        }
    }

    printf("\nProcess\tBurst Time\tWaiting Time\tTurnaround Time\n");
    for (i = 0; i < n; i++) {
        totwtime += p[i].wtime = tbm;
        tbm += p[i].btime;
        printf("%d\t\t%d\t\t%d\t\t%d\n", p[i].pno, p[i].btime, p[i].wtime, p[i].wtime + p[i].btime);
    }

    totttime = tbm;

    printf("\nTotal Waiting Time: %d\n", totwtime);
    printf("Average Waiting Time: %f\n", (float)totwtime / n);
    printf("Total Turnaround Time: %d\n", totttime);
    printf("Average Turnaround Time: %f\n", (float)totttime / n);

    free(p);

    return 0;
}
```

OUTPUT:

```
PRIORITY SCHEDULING.

Enter the number of processes:
3
Enter the burst time and priority for each process:
Process 1:
4 2
Process 2:
3 1
Process 3:
8 3

Process    Burst Time    Waiting Time    Turnaround Time
   1           5              0                5
   2           3              5                8
   3           8              8               16

Total Waiting Time: 13
Average Waiting Time: 4.333333
Total Turnaround Time: 29
Average Turnaround Time: 9.666667
```

Ex.No:4.b Date:	CPU SCHEDULING ALGORITHMS
	ROUND ROBIN SCHEDULING

AIM:

To write a C program for implementation of Round Robin scheduling algorithms.

ALGORITHM:

Step 1: Inside the structure declare the variables.
Step 2: Declare the variable i,j as integer, totwtime and totttime is equal tozero.
Step 3: Get the value of „n' assign p and allocate the memory.
Step 4: Inside the for loop get the value of burst time and priority and read the time quantum.
Step 5: Assign wtime as zero.
Step 6: Check p[i].pri is greater than p[j].pri .
Step 7: Calculate the total of burst time and waiting time and assign as turnaroundtime.
Step 8: Stop the program.

PROGRAM:

```
#include<stdio.h>
#include<stdlib.h>

struct rr {
    int pno, btime, sbtime, wtime, lst;
} p[10];

int main() {
    int pp = -1, ts, flag, count, ptm = 0, i, n, twt = 0, totttime = 0;

    printf("\nRound Robin Scheduling.\n");
    printf("Enter the number of processes: ");
    scanf("%d", &n);
    printf("Enter the time slice: ");
    scanf("%d", &ts);
    printf("Enter the burst time:\n");
    for(i = 0; i < n; i++) {
        printf("Process %d: ", i + 1);
        scanf("%d", &p[i].btime);
        p[i].wtime = p[i].lst = 0;
        p[i].pno = i + 1;
        p[i].sbtime = p[i].btime;
    }
    printf("Scheduling...\n");

    do {
        flag = 0;
        for(i = 0; i < n; i++) {
            count = p[i].btime;
            if(count > 0) {
                flag = -1;
                count = (count >= ts) ? ts : count;
                printf("\nProcess %d from %d to ", p[i].pno, ptm);
                ptm += count;
                printf("%d", ptm);
                p[i].btime -= count;
                if(pp != i) {
                    pp = i;
                    p[i].wtime += ptm - p[i].lst - count;
                    p[i].lst = ptm;
                }
            }
        }
    } while(flag == -1);
    printf("\n\nProcess\tWaiting Time\n");
    for(i = 0; i < n; i++) {
        twt += p[i].wtime;
        printf("%d\t\t%d\n", p[i].pno, p[i].wtime);
    }

    totttime = ptm;
    printf("\nTotal Waiting Time: %d\n", twt);
    printf("Average Waiting Time: %f\n", (float)twt / n);
    printf("Total Turnaround Time: %d\n", totttime);
    printf("Average Turnaround Time: %f\n", (float)totttime / n);

    return 0;
}
```

OUTPUT:

```
Round Robin Scheduling
Enter the number of processes: 2
Enter the time slice: 3
Enter the burst time:
Process 1: 5
Process 2: 6

Scheduling...
Process 1: 0 to 3
Process 2: 3 to 6
Process 1: 6 to 8
Process 2: 8 to 11

Process       Burst Time      Wait Time
1                    5              3
2                     6              5
```

```
Total Wait Time: 8
Average Wait Time: 4.000000
```

Ex.No:4.c	CPU SCHEDULING ALGORITHMS
Date:	FCFS

AIM:
To write a C program for implementation of FCFS scheduling algorithms.

ALGORITHM:

Step 1: Inside the structure declare the variables.
Step 2: Declare the variable i,j as integer,totwtime and totttime is equal tozero.
Step 3: Get the value of „n' assign pid as I and get the value of p[i].btime.
Step 4: Assign p[0] wtime as zero and tot time as btime and inside the loop calculate wait timeand turnaround time.
Step 5: Calculate total wait time and total turnaround time by dividing by total number of process.
Step 6: Print total wait time and total turnaround time.
Step 7: Stop the program.

PROGRAM:

```
#include <stdio.h>
#include <stdlib.h>

struct fcfs {
    int pid;
    int btime;
    int wtime;
    int ttime;
} p[10];

int main() {
    int i, n;
    int totwtime = 0, totttime = 0;

    printf("\nFCFS Scheduling...\n");
    printf("Enter the number of processes: ");
    scanf("%d", &n);
    for (i = 0; i < n; i++) {
        p[i].pid = i + 1;
        printf("\nEnter burst time of process %d: ", p[i].pid);
        scanf("%d", &p[i].btime);
    }
    p[0].wtime = 0;
    p[0].ttime = p[0].btime;
    totttime += p[0].ttime;

    for (i = 1; i < n; i++) {
        p[i].wtime = p[i - 1].wtime + p[i - 1].btime;
        p[i].ttime = p[i].wtime + p[i].btime; // Corrected here
        totttime += p[i].ttime;
        totwtime += p[i].wtime;
    }
    printf("\nProcess\tWaiting Time\tTurnaround Time\n");
    for (i = 0; i < n; i++) {
        printf("%d\t%d\t\t%d\n", p[i].pid, p[i].wtime, p[i].ttime);
    }

    printf("\nTotal Waiting Time: %d\n", totwtime);
    printf("Average Waiting Time: %f\n", (float)totwtime / n);
    printf("Total Turnaround Time: %d\n", totttime);
    printf("Average Turnaround Time: %f\n", (float)totttime / n);

    return 0;
}
```

OUTPUT:

```
FCFS scheduling...
Enter the number of processes: 3

 Burst time of process 1: 4

 Burst time of process 2: 3

 Burst time of process 3: 6

Process   Burst time    Waiting time    Turnaround time
1               4             0              4
 2              3              4            7
3               6              7            13

Total waiting time: 11
Average waiting time: 3.666667
Total turnaround time: 24
Average turnaround time: 8.000000
```

Ex.No:4.d Date:	CPU SCHEDULING ALGORITHMS
	SJFS SCHEDULING

AIM:
To write a C program for implementation of SJFS scheduling algorithms.

ALGORITHM:
Step 1: Inside the structure declare the variables.
Step 2: Declare the variable i, j as integer, totwtime and totttime is equal to zero.
Step 3: Get the value of
„n' assign pid as I and get the value of p[i].btime.
Step 4: Assign p[0] wtime as zero and tot time as btime and inside the loop calculate wait time andturnaround time.
Step 5: Calculate total wait time and total turnaround time by dividing by total number of process.
Step 6: Print total wait time and total turnaround time.
Step 7: Stop the program.

PROGRAM:
```
#include <stdio.h>
#include <stdlib.h>
typedef struct {
    int pid;
    int btime;
    int wtime;
} sp;
int main() {
    int i, j, n, tbm = 0, totwtime = 0, totttime, t;
    printf("\nSJF Scheduling..\n");
    printf("Enter the number of processors: ");
    scanf("%d", &n);
    sp *p = (sp *)malloc(n * sizeof(sp));
    printf("\nEnter the burst time:\n");
    for (i = 0; i < n; i++) {
        printf("Process %d: ", i + 1);
        scanf("%d", &p[i].btime);
        p[i].pid = i + 1;
        p[i].wtime = 0;
    }
    for (i = 0; i < n; i++)
        for (j = i + 1; j < n; j++)
            if (p[i].btime > p[j].btime) {
                sp temp = p[i];
                p[i] = p[j];
                p[j] = temp;
            }
    printf("\nProcess Scheduling:\n");
    printf("\nProcess \tBurst Time \tWaiting Time\n");
    for (i = 0; i < n; i++) {
        totwtime += p[i].wtime = tbm;
        tbm += p[i].btime;
        printf("%d\t\t%d\t\t%d\n", p[i].pid, p[i].btime, p[i].wtime);
    }
    totttime = tbm;
    printf("\nTotal Waiting Time: %d", totwtime);
    printf("\nAverage Waiting Time: %f", (float)totwtime / n);
    printf("\nTotal Turnaround Time: %d", totttime);
    printf("\nAverage Turnaround Time: %f\n", (float)totttime / n);

    free(p);
    return 0;
}
```

OUTPUT:

```
SJF scheduling ..
enter the no of processor:2

 enter the burst time:
 process 1      5

 process 2      6

 process scheduling

 process        burst time       waiting time

1                          5              0
2                          6              5
 total waiting time :5
 average waiting time :2.500000
 total turn around time :11
 average turn around time: :5.500000
```

RESULT:

Thus the programs implementing CPU scheduling algorithms are executed successfully.

Ex.No:5 Date:	IPC USING SHARED MEMORY

AIM:
To write a c program to implement IPC using shared memory.

ALGORITHM:

Step 1: Start the process.
Step 2: Declare the segment size.
Step 3: Create the shared memory
Step 4: Read the data from the shared memory
Step 5: Write the data to the shared memory.
Step 6: Edit the data.
Step 7: Stop the process.

PROGRAM:

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <sys/shm.h>
#include <sys/types.h>
#include <sys/ipc.h>

#define SEGSIZE 100

int main(int argc, char *argv[]) {
  int shmid;
  key_t key;
  char *segptr;
  char buff[] = "Hello World.";

  key = ftok(".", 's');
  if ((shmid = shmget(key, SEGSIZE, IPC_CREAT | IPC_EXCL | 0666)) == -1) {
    if ((shmid = shmget(key, SEGSIZE, 0)) == -1) {
      perror("shmget");
      exit(1);
    }
  }

  printf("Creating a new shared memory segment\n");
  printf("SHMID: %d\n", shmid);
  system("ipcs -m");

  if ((segptr = (char *)shmat(shmid, 0, 0)) == (char *)-1) {
    perror("shmat");
    exit(1);
  }

  printf("Writing data to shared memory…\n");
  strcpy(segptr, buff);
  printf("DONE\n");

  printf("Reading data from shared memory…\n");
  printf("DATA: %s\n", segptr);
  printf("DONE\n");

  printf("Removing shared memory segment…\n");
  if (shmctl(shmid, IPC_RMID, 0) == -1) {
    printf("Can’t Remove Shared memory Segment…\n");
  } else {
    printf("Removed Successfully\n");
  }

  return 0;
}
```

OUTPUT:

```
Creating a new shared memory seg
SHMID:0

------ Shared Memory Segments --------
key          shmid owner  perms  bytes nattch status
0x7302d287    0     root    666    100     0
Writing data to shared memory…
DONE
Reading data from shared memory…
DATA:-Hello World.
DONE
Removing shared memory Segment…
Removed Successfully
```

RESULT:

Thus the c program to implement IPC using shared memory is executed successfully.

Ex.No:6 **Date:**	**PRODUCER CONSUMER PROBLEM USING SEMAPHORES**

AIM:
To write a C-program to implement the producer – consumer problem using semaphores.

ALGORITHM:

Step 1: Start the program.
Step 2: Declare the required variables.
Step 3: Initialize the buffer size and get maximum item you want to produce.
Step 4: Get the option, which you want to do either producer, consumer or exit fromthe operation.
Step 5: If you select the producer, check the buffer size if it is full the producer should not produce the item or otherwise produce the item and increase the value buffersize.
Step 6: If you select the consumer, check the buffer size if it is empty the consumer should notconsume the item or otherwise consume the item and decrease the value of buffer size.
Step 7: If you select exit come out of the program.
Step 8: Stop the program.

PROGRAM:

```
#include<stdio.h>
#include<stdlib.h>

int mutex = 1, full = 0, empty = 3, x = 0;

void producer();
void consumer();
int wait(int);
int signal(int);

int main() {
    int n;
    printf("\n1.PRODUCER\n2.CONSUMER\n3.EXIT\n");
    while(1) {
        printf("\nENTER YOUR CHOICE\n");
        scanf("%d", &n);
        switch(n) {
            case 1:
                if((mutex == 1) && (empty != 0))
                    producer();
                else
                    printf("BUFFER IS FULL\n");
                break;
            case 2:
                if((mutex == 1) && (full != 0))
                    consumer();
                else
                    printf("BUFFER IS EMPTY\n");
                break;
            case 3:
                exit(0);
                break;
            }
        }
        return 0;
    }

    int wait(int s) {
        return (--s);
    }

    int signal(int s) {
        return (++s);
    }

    void producer() {
        mutex = wait(mutex);
        full = signal(full);
        empty = wait(empty);
        x++;
        printf("\nproducer produces the item %d\n", x);
        mutex = signal(mutex);
    }

    void consumer() {
        mutex = wait(mutex);
        full = wait(full);
        empty = signal(empty);
        printf("\nconsumer consumes item %d\n", x);
        x--;
        mutex = signal(mutex);
    }
```

OUTPUT:-

```
 1.PRODUCER
2.CONSUMER
3.EXIT
ENTER YOUR CHOICE: 2
BUFFER IS EMPTY
```

RESULT:

Thus the C-program to implement the producer – consumer problem using semaphores is executed successfully.

Ex.No:7 Date:	BANKERSALGORITHM FOR DEADLOCK AVOIDANCE

AIM:
To write a C program to implement banker's algorithm for deadlock avoidance.

ALGORITHM:

Step-1: Start the program.
Step-2: Declare the memory for the process.
Step-3: Read the number of process, resources, allocation matrix and available matrix.
Step-4: Compare each and every process using the banker's algorithm.
Step-5: If the process is in safe state then it is a not a deadlock process otherwise it is a deadlock process
Step-6:Produce the result of state of process
Step-7: Stop the program

PROGRAM:

```
#include <stdio.h>

int max[100][100];
int alloc[100][100];
int need[100][100];
int avail[100];
int n, r;

void input();
void show();
void cal();

int main() {
    printf("********** Banker's Algo ************\n");
    input();
    show();
    cal();
    return 0;
}

void input() {
    printf("Enter the no of Processes\t");
    scanf("%d", &n);
    printf("Enter the no of resources instances\t");
    scanf("%d", &r);
    printf("Enter the Max Matrix\n");
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < r; j++) {
            scanf("%d", &max[i][j]);

        }
    }
    printf("Enter the Allocation Matrix\n");
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < r; j++) {
            scanf("%d", &alloc[i][j]);
        }
    }
    printf("Enter the available Resources\n");
    for (int j = 0; j < r; j++) {
        scanf("%d", &avail[j]);
    }
}

void show() {
    printf("Process\t Allocation\t Max\t Available\n");
    for (int i = 0; i < n; i++) {
        printf("P%d\t", i + 1);
        for (int j = 0; j < r; j++) {
            printf("%d ", alloc[i][j]);
        }
        printf("\t");
        for (int j = 0; j < r; j++) {
            printf("%d ", max[i][j]);
        }
        printf("\t");
        if (i == 0) {
            for (int j = 0; j < r; j++) {
                printf("%d ", avail[j]);
            }
        }
        printf("\n");
    }
}

void cal() {
    int finish[100];
    int flag = 1;
    int c1 = 0;

    for (int i = 0; i < n; i++) {
        finish[i] = 0;
    }

    // Find need matrix
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < r; j++) {
            need[i][j] = max[i][j] - alloc[i][j];

        }
    }

    printf("\n");

    while (flag) {
        flag = 0;
        for (int i = 0; i < n; i++) {
            int c = 0;
            for (int j = 0; j < r; j++) {
                if ((finish[i] == 0) && (need[i][j] <= avail[j])) {
                    c++;
                    if (c == r) {
                        for (int k = 0; k < r; k++) {
                            avail[k] += alloc[i][j];
                        }
                        finish[i] = 1;
                        flag = 1;
                        printf("P%d->", i);
                        if (finish[i] == 1) {
                            i = n;
                        }
                    }
                }
            }
        }
    }

    for (int i = 0; i < n; i++) {
        if (finish[i] == 1) {
            c1++;
        } else {
            printf("P%d->", i);
        }
    }

    if (c1 == n) {
        printf("\nThe system is in safe state\n");
    } else {
        printf("\nProcess are in dead lock\n");
        printf("System is in unsafe state\n");
    }
}
```

OUTPUT:

```
********** Banker's Algo ************
Enter the no of Processes    5
Enter the no of resources instances    3
Enter the Max Matrix
7 5 3
3 2 2
9 0 2
2 2 2
6 3 3
Enter the Allocation Matrix
0 1 0
2 0 0
3 0 2
2 1 1
0 0 2
Enter the available Resources
3 3 2
```

RESULT:

Thus the C program to implement banker's algorithm for deadlock avoidance is executed successfully.

Ex.No:8 Date:	IMPLEMENTATION OF DEADLOCK DETECTION ALGORITHM

AIM:

To write a C program to implement algorithm for deadlock detection.

ALGORITHM:

Step-1: Start the program.
Step-2: Declare the memory for the process.
Step-3: Read the number of process, resources, allocation matrix and available matrix.
Step-4: Compare each and every process using the banker's algorithm.
Step-5: If the process is in safe state then it is a not a deadlock process otherwise it is a deadlock process
Step-6:Produce the result of state of process
Step-7: Stop the program

PROGRAM:

```
#include <stdio.h>
#include <conio.h>

int max[100][100], alloc[100][100], need[100][100], avail[100];
int n, r;

void input();
void show();
void cal();

int main() {
   printf("********** Deadlock Detection Algorithm ************\n");
   input();
   show();
   cal();
   getch();
   return 0;
}

void input() {
   int i, j;
   printf("Enter the number of Processes: ");
   scanf("%d", &n);
   printf("Enter the number of resource instances: ");
   scanf("%d", &r);
   printf("Enter the Max Matrix:\n");
   for(i = 0; i < n; i++) {
       for(j = 0; j < r; j++) {
           scanf("%d", &max[i][j]);
       }
   }
   printf("Enter the Allocation Matrix:\n");
   for(i = 0; i < n; i++) {
       for(j = 0; j < r; j++) {
           scanf("%d", &alloc[i][j]);
       }
   }
   printf("Enter the available Resources:\n");
   for(j = 0; j < r; j++) {
       scanf("%d", &avail[j]);
   }
}

void show() {
   int i, j;
   printf("Process\t Allocation\t Max\t Available\n");
   for(i = 0; i < n; i++) {
       printf("P%d\t ", i + 1);
       for(j = 0; j < r; j++) {
           printf("%d ", alloc[i][j]);
       }
       printf("\t");
       for(j = 0; j < r; j++) {
           printf("%d ", max[i][j]);
       }
       printf("\t");
       if(i == 0) {
           for(j = 0; j < r; j++) printf("%d ", avail[j]);
       }
       printf("\n");
   }
}

void cal() {
   int finish[100], flag = 1, k, i, j;
   int dead[100];

   for(i = 0; i < n; i++) {
       finish[i] = 0;
   }

   // Calculate need matrix
   for(i = 0; i < n; i++) {
       for(j = 0; j < r; j++) {
           need[i][j] = max[i][j] - alloc[i][j];
       }
   }

   // Detect deadlock
   while(flag) {
       flag = 0;
       for(i = 0; i < n; i++) {
           int c = 0;
           for(j = 0; j < r; j++) {
               if((finish[i] == 0) && (need[i][j] <= avail[j])) {
                   c++;
                   if(c == r) {
                       for(k = 0; k < r; k++) {
                           avail[k] += alloc[i][j];
                           finish[i] = 1;
                           flag = 1;
                       }
                   }
               }
           }
       }
   }

   // Find deadlocked processes
   j = 0;
   flag = 0;
   for(i = 0; i < n; i++) {
       if(finish[i] == 0) {
           dead[j] = i;
           j++;
           flag = 1;
       }
   }

   // Output deadlock status
   if(flag == 1) {
       printf("\n\nSystem is in Deadlock and the Deadlocked processes are:\n");
       for(i = 0; i < j; i++) {
           printf("P%d\t", dead[i] + 1);
       }
       printf("\n");
   } else {
       printf("\nNo Deadlock Occurs\n");
   }
}
```

OUTPUT:

```
********** Deadlock Detection Algo ************
Enter the no of Processes    4
Enter the no of resource instances 3
Enter the Max Matrix
3 2 2
1 2 3
1 3 2
3 3 3
Enter the Allocation Matrix
1 0 2
1 1 0
0 0 1
2 1 1
Enter the available Resources
1 1 1
Process       Allocation     Max          Available
P1                1 0 2       3 2 2           1 1 1
P2                1 1 0       1 2 3     (available resources)
P3                0 0 1       1 3 2
P4                2 1 1       3 3 3
System is in Deadlock and the Deadlocked processes are:
P1  P2  P4
```

RESULT:

Thus the C program to implement algorithm for deadlock detection is executed successfully.

Ex.No:9 Date:	THREADING

AIM:
To write a c program to implement Threading.

ALGORITHM:

Step 1: Start the process
Step 2: Declare process thread, thread-id.
Step 3: Read the process thread and thread state.
Step 4: Check the process thread equals to thread-id by using if condition.
Step 5: Check the error state of the thread.
Step 6: Display the completed thread process.
Step 7: Stop the process

PROGRAM:

```
#include <stdio.h>
#include <string.h>
#include <pthread.h>
#include <stdlib.h>
#include <unistd.h>

pthread_t tid[2];

void *doSomeThing(void *arg) {
   unsigned long i = 0;
   pthread_t id = pthread_self();

   if (pthread_equal(id, tid[0])) {
      printf("\nFirst thread processing\n");
   } else {
      printf("\nSecond thread processing\n");
   }

   for (i = 0; i < (0xFFFFFFFF); i++);

   return NULL;
}

int main(void) {
   int i = 0;
   int err;

   while (i < 2) {
      err = pthread_create(&(tid[i]), NULL, &doSomeThing, NULL);
      if (err != 0)
```

```
            printf("\ncan't create thread :[%s]", strerror(err));

        else
            printf("\nThread created successfully\n");
        i++;
    }

    sleep(5);
    return 0;
}
```

OUTPUT:

```
Thread created successfully
 Thread created successfully

 First thread processing
 Second thread processing
```

RESULT:

Thus the c program to implement Threading is executed successfully.

Ex.No:10 **Date:**	**PAGING TECHNIQUE OF MEMORY MANAGEMENT**

AIM:
To write a c program to implement Paging technique for memory management.

ALGORITHM:

Step 1: Start the process
Step 2: Declare page number, page table, frame number and process size.
Step 3: Read the process size, total number of pages
Step 4: Read the relative address
Step 5: Calculate the physical address
Step 6: Display the address
Step 7: Stop the process

PROGRAM:

```
#include <stdio.h>
#include <string.h>
#include <pthread.h>
#include <stdlib.h>
#include <unistd.h>

pthread_t tid[2];
int counter;
pthread_mutex_t lock;

void* doSomeThing(void* arg) {
    pthread_mutex_lock(&lock);
    unsigned long i = 0;
    counter += 1;
    printf("\n Job %d started\n", counter);
    for (i = 0; i < (0xFFFFFFFF); i++);
    printf("\n Job %d finished\n", counter);
    pthread_mutex_unlock(&lock);
    return NULL;
}

int main(void) {
    int i = 0;
    int err;

    if (pthread_mutex_init(&lock, NULL) != 0) {
        printf("\n mutex init failed\n");
        return 1;
    }
    while (i < 2) {
        err = pthread_create(&(tid[i]), NULL, &doSomeThing, NULL);
        if (err != 0)
            printf("\ncan't create thread :[%s]", strerror(err));
        i++;
    }

    pthread_join(tid[0], NULL);
    pthread_join(tid[1], NULL);
    pthread_mutex_destroy(&lock);

    return 0;
```

OUTPUT:

```
Job 1 started
 Job 1 finished

 Job 2 started
 Job 2 finished
```

RESULT:

Thus the c program to implement Paging technique for memory management is executed successfully.

Ex.No:11.a Date:	MEMORY ALLOCATION METHODS FOR FIXED PARTITION
	FIRST FIT

AIM:
To write a C program for implementation memory allocation methods for fixed partition using first fit.

ALGORITHM:

Step 1:Define the max as 25.
Step 2: Declare the variable frag[max],b[max],f[max],i,j,nb,nf,temp, highest=0, bf[max],ff[max].
Step 3: Getthe number of blocks,files,size of the blocks using for loop.
Step 4: In for loop check bf[j]!=1, if so temp=b[j]-f[i]
Step 5: Check highest<temp,if so assign ff[i]=j,highest=tempStep 6: Assign frag[i]=highest, bf[ff[i]]=1,highest=0
Step 7: Repeat step 4 to step 6.
Step 8: Print file no, size, block no, size and fragment.
Step 9: Stop the program.

PROGRAM:
```
#include<stdio.h>
#include<conio.h>
#define max 25

void main()
{
   int frag[max], b[max], f[max], bf[max], ff[max];
   int i, j, nb, nf, temp, highest = 0;

   clrscr();

   printf("\n\tMemory Management Scheme - Worst Fit\n");
   printf("Enter the number of blocks: ");
   scanf("%d", &nb);
   printf("Enter the number of files: ");
   scanf("%d", &nf);

   printf("\nEnter the size of the blocks:\n");
   for(i = 1; i <= nb; i++)
   {
      printf("Block %d: ", i);
      scanf("%d", &b[i]);
   }

   printf("Enter the size of the files:\n");
   for(i = 1; i <= nf; i++)
   {
      printf("File %d: ", i);
```

```
        scanf("%d", &f[i]);
    }

    for(i = 1; i <= nf; i++)
    {
        for(j = 1; j <= nb; j++)
        {
            if(bf[j] != 1) // if bf[j] is not allocated
            {
                temp = b[j] - f[i];
                if(temp >= 0)
                {
                    if(highest < temp)
                    {
                        ff[i] = j;
                        highest = temp;
                    }
                }
            }
        }

        frag[i] = highest;
        bf[ff[i]] = 1;
        highest = 0;
    }

    printf("\nFile_no:\tFile_size:\tBlock_no:\tBlock_size:\tFragment\n");
    for(i = 1; i <= nf; i++)
    {
        printf("%d\t\t%d\t\t%d\t\t%d\t\t%d\n", i, f[i], ff[i], b[ff[i]], frag[i]);
    }

    getch();
}
```

OUTPUT:

```
Memory Management Scheme -First Fit
Enter the number of blocks: 3
Enter the number of files: 2
Enter the size of the blocks:
Block 1: 50
Block 2: 100
Block 3: 200
Enter the size of the files:
File 1: 75
File 2: 150

File_no File_size     Block_no       Block_size      Fragement
1            75             3                    200           125
```

2	150	3	200	50

Ex.No:11.b Date:	MEMORY ALLOCATION METHODS FOR FIXED PARTITION
	WORST FIT

AIM:
To write a C program for implementation of FCFS and SJF scheduling algorithms.

ALGORITHM:

Step 1:Define the max as 25.
Step 2: Declare the variable frag[max],b[max],f[max],i,j,nb,nf,temp, highest=0, bf[max],ff[max].
Step 3: Get the number of blocks,files,size of the blocks using for loop.
Step 4: In for loop check bf[j]!=1, if so temp=b[j]-f[i]
Step 5: Check temp>=0,if so assign ff[i]=j break the for loop.
Step 6: Assign frag[i]=temp,bf[ff[i]]=1;
Step 7: Repeat step 4 to step 6.
Step 8: Print file no,size,block no,size and fragment.
Step 9: Stop the program.

PROGRAM:

```
#include<stdio.h>
#define max 25

void main() {
   int frag[max], b[max], f[max], i, j, nb, nf, temp;
   static int bf[max], ff[max];

   printf("\n\tMemory Management Scheme - Worst Fit\n");
   printf("Enter the number of blocks: ");
   scanf("%d", &nb);
   printf("Enter the number of files: ");
   scanf("%d", &nf);

   printf("\nEnter the size of the blocks:\n");
   for (i = 1; i <= nb; i++) {
      printf("Block %d: ", i);
      scanf("%d", &b[i]);
   }
   printf("\nEnter the size of the files:\n");
   for (i = 1; i <= nf; i++) {
      printf("File %d: ", i);
      scanf("%d", &f[i]);
   }

   for (i = 1; i <= nf; i++) {
      for (j = 1; j <= nb; j++) {
            if (bf[j] != 1) {

                temp = b[j] - f[i];
                if (temp >= 0) {
                    ff[i] = j;
                    break;
                }
            }
        }
        frag[i] = temp;
        bf[ff[i]] = 1;

        printf("\nFile_no:\tFile_size :\tBlock_no:\tBlock_size:\tFragment\n");
        for (i = 1; i <= nf; i++)
            printf("%d\t\t%d\t\t%d\t\t%d\t\t%d\n", i, f[i], ff[i], b[ff[i]], frag[i]);
        getchar(); // Pause for user input
    }
}
```

OUTPUT:

```
Memory Management Scheme - Worst Fit
Enter the number of blocks: 5
Enter the number of files: 3

Enter the size of the blocks:-
Block 1: 150
Block 2: 350
Block 3: 50
Block 4: 100
Block 5: 200
Enter the size of the files :-
File 1: 130
File 2: 200
File 3: 300

File_no:     File_size :    Block_no:     Block_size:    Fragement
1            130            1             150            20
2            200            2             350            150
3            300            5             200            -100
```

Ex.No:11.c Date:	MEMORY ALLOCATION METHODS FOR FIXED PARTITION
	BEST FIT

AIM:
To write a C program for implementation of FCFS and SJF scheduling algorithms.

ALGORITHM:

Step 1:Definethe max as 25.
Step 2: Declare the variable frag[max],b[max],f[max],i,j,nb,nf,temp, highest=0, bf[max],ff[max].
Step 3: Get the number of blocks,files,size of the blocks using for loop.
Step 4: In for loop check bf[j]!=1, if so temp=b[j]-f[i]
Step 5: Check lowest>temp,if so assign ff[i]=j,highest=temp
Step 6: Assign frag[i]=lowest,
bf[ff[i]]=1,lowest=10000
Step 7: Repeat step 4 to step6.
Step 8: Print file no,size,block no,size and fragment.
Step 9: Stop the program.

PROGRAM:
```
#include<stdio.h>
#include<stdlib.h>

#define max 25

int main() {
   int frag[max], b[max], f[max], bf[max], ff[max];
   int i, j, nb, nf, temp, lowest;

   printf("Enter the number of blocks: ");
   scanf("%d", &nb);
   printf("Enter the number of files: ");
   scanf("%d", &nf);

   printf("Enter the size of the blocks:\n");
   for(i = 1; i <= nb; i++) {
      printf("Block %d: ", i);
      scanf("%d", &b[i]);
   }
   printf("Enter the size of the files:\n");
   for(i = 1; i <= nf; i++) {
      printf("File %d: ", i);
      scanf("%d", &f[i]);
   }
```

```
    for(i = 1; i <= nf; i++) {

        lowest = 10000;
        for(j = 1; j <= nb; j++) {
            if(bf[j] != 1) {
                temp = b[j] - f[i];
                if(temp >= 0 && lowest > temp) {
                    ff[i] = j;
                    lowest = temp;
                    frag[i] = lowest;
                    bf[ff[i]] = 1;
                }
            }
        }
    }

    printf("\nFile No\tFile Size\tBlock No\tBlock Size\tFragment\n");
    for(i = 1; i <= nf && ff[i] != 0; i++) {
        printf("%d\t\t%d\t\t%d\t\t%d\t\t%d\n", i, f[i], ff[i], b[ff[i]], frag[i]);
    }

    return 0;
}
```

OUTPUT:

```
Enter the number of blocks: 4
Enter the number of files: 3

Enter the size of the blocks:-
Block 1: 100
Block 2: 200
Block 3: 300
Block 4: 400

Enter the size of the files:-
File 1: 150
File 2: 250
File 3: 350

File No      File Size      Block No      Block Size      Fragment
1            150            2             200             50
2            250            3             300             50
3            350            4             400             50
```

RESULT:

Thus the C program implementing memory allocation methods for fixed partitions is executed successfully.

Ex.No:12.a	**PAGE REPLACEMENT ALGORITHMS**
Date:	**FIFO**

AIM:

To write a C program for implementation of FIFO page replacement algorithm.

ALGORITHM:

Step 1: Start the program.
Step 2: Declare the necessary variables.
Step 3: Enter the number of frames.
Step 4: Enter the reference string end with zero.
Step 5: FIFO page replacement selects the page that has been in memory the longest timeand when the page must be replaced the oldest page is chosen.
Step 6: When apage is brought into memory, it is inserted at the tail of the queue.
Step 7: Initially all the three frames are empty.
Step 8: The page fault range increases as the no of allocated frames alsoincreases.
Step 9: Print the total number of page faults.
Step 10: Stop the program.

PROGRAM:

```
#include<stdio.h>
#include<stdlib.h>

int main() {
    int i = 0, j = 0, k = 0, i1 = 0, m, n, rs[30], flag = 1, p[30];
    system("clear");
    printf("FIFO page replacement algorithm....\n");
    printf("Enter the number of frames: ");
    scanf("%d", &n);
    printf("Enter the reference string (enter 0 to end): ");
    while (1) {
        scanf("%d", &rs[i]);
        if (rs[i] == 0)
            break;
        i++;
    }
    m = i;
    for (j = 0; j < n; j++)
        p[j] = 0;
    for (i = 0; i < m; i++) {
        flag = 1;
        for (j = 0; j < n; j++)
            if (p[j] == rs[i]) {
                printf("Data already in page...\n");
                flag = 0;
                break;
            }
        if (flag == 1) {
            p[i1] = rs[i];
            i1++;
            k++;
            if (i1 == n)
                i1 = 0;
            for (j = 0; j < n; j++) {
                printf("\nPage %d: %d", j + 1, p[j]);
                if (p[j] == rs[i])
                    printf("*");
            }
            printf("\n\nTotal number of page faults = %d\n", k);
        }
    }
    return 0;
}
```

OUTPUT:

```
FIFO page replacement algorithm....
enter the no. of frames:3
enter the reference string:1 2 3 4 1 2 5 1 2 3 4 5 6 0

 page 1:1
 page 2:0
 page 3:0

 page 1:1
 page 2:2
 page 3:0

 page 1:1
 page 2:2
 page 3:3

 page 1:4
 page 2:2
 page 3:3
data already in page....
data already in page....
 page 1:4
 page 2:5
 page 3:3
data already in page....

data already in page....

data already in page....

data already in page....

 page 1:4
 page 2:5
 page 3:6
```

```
total no page faults=6
```

Ex.No:12.b	PAGE REPLACEMENT ALGORITHMS
Date:	LRU

AIM:
To write a c program to implement LRU page replacement algorithm.

ALGORITHM:

Step 1: Start the process
Step 2: Declare the size
Step 3: Get the number of pages to be inserted
Step 4: Get the value
Step 5: Declare counter and stack
Step 6: Select the least recently used page by counter value
Step 7: Stack them according the selection.
Step 8: Display the values
Step 9: Stop the process

PROGRAM:

```
#include <stdio.h>

int main() {
    int q[20], p[50], c = 0, c1, d, f, i, j, k = 0, n, r, t, b[20], c2[20];
    printf("Enter no of pages: ");
    scanf("%d", &n);
    printf("Enter the reference string: ");
    for (i = 0; i < n; i++)
        scanf("%d", &p[i]);
    printf("Enter no of frames: ");
    scanf("%d", &f);
    q[k] = p[k];
    printf("\n\t%d\n", q[k]);
    c++;
    k++;
    for (i = 1; i < n; i++) {
        c1 = 0;
        for (j = 0; j < f; j++) {
            if (p[i] != q[j])
                c1++;
        }
        if (c1 == f) {
            c++;
            if (k < f) {
                q[k] = p[i];
                k++;
                for (j = 0; j < k; j++)
                    printf("\t%d", q[j]);
                printf("\n");
            } else {
                for (r = 0; r < f; r++) {
                    c2[r] = 0;
                    for (j = i - 1; j >= 0; j--) {
                        if (q[r] != p[j])
                            c2[r]++;
                        else
                            break;
                    }
                }
                for (r = 0; r < f; r++)
                    b[r] = c2[r];
                for (r = 0; r < f; r++) {
                    for (j = r; j < f; j++) {
                        if (b[r] < b[j]) {
                            t = b[r];
                            b[r] = b[j];
                            b[j] = t;
                        }
                    }
                }
                for (r = 0; r < f; r++) {
                    if (c2[r] == b[0])
                        q[r] = p[i];
                    printf("\t%d", q[r]);
                }
                printf("\n");
            }
        }
    }
    printf("\nThe no of page faults is %d\n", c);
    return 0;
}
```

OUTPUT:

```
Enter no of pages: 7
Enter the reference string: 2 3 2 1 5 2 4
Enter no of frames: 3

    2
    2   3
    2   3   1
    5   3   1
    5   2   1
    5   2   4

Total number of page faults is 5
```

Ex.No:12.c	PAGE REPLACEMENT ALGORITHMS
Date:	LFU

AIM:

To write C program to implement LFU page replacement algorithm.

ALGORITHM:

Step 1: Start the process
Step 2: Declare the size
Step 3: Get the number of pages to be inserted
Step 4: Get the value
Step 5: Declare counter and stack
Step 6: Select the least frequently used page by counter value
Step 7: Stack them according the selection.
Step 8: Display the values
Step 9: Stop the process

PROGRAM:

```
#include<stdio.h>

int main() {
    int f, p;
    int pages[50], frame[10], hit = 0, count[50], time[50];
    int i, j, page, flag, least, minTime, temp;

    printf("Enter no of frames: ");
    scanf("%d", &f);
    printf("Enter no of pages: ");
    scanf("%d", &p);

    for (i = 0; i < f; i++) {
        frame[i] = -1;
    }

    for (i = 0; i < 50; i++) {
        count[i] = 0;
    }

    printf("Enter page numbers: \n");
    for (i = 0; i < p; i++) {
        scanf("%d", &pages[i]);
    }
    printf("\n");

    for (i = 0; i < p; i++) {
        count[pages[i]]++;
        time[pages[i]] = i;
        flag = 1;
        least = frame[0];

        for (j = 0; j < f; j++) {
            if (frame[j] == -1 || frame[j] == pages[i]) {
                if (frame[j] != -1) {
                    hit++;
                }
                flag = 0;
                frame[j] = pages[i];
                break;
            }

            if (count[least] > count[frame[j]]) {
                least = frame[j];
            }
        }

        if (flag) {
            minTime = 50;
            for (j = 0; j < f; j++) {
                if (count[frame[j]] == count[least] && time[frame[j]] < minTime) {
                    temp = j;
                    minTime = time[frame[j]];
                }
            }
            count[frame[temp]] = 0;
            frame[temp] = pages[i];
        }

        for (j = 0; j < f; j++) {
            printf("%d ", frame[j]);
        }
        printf("\n");
    }

    printf("Page hit = %d\n", hit);
    return 0;
}
```

OUTPUT:

```
Enter no of frames : 4
Enter no of pages : 15
Enter page no :
1
2
3
4
1
2
5
1
2
3
4
5
6
3
4
1 -1 -1 -1
1 2 -1 -1
1 2 3 -1
1 2 3 4
- 2 3 4
- 2 5 4
- 1 5 4
- 1 2 4
- - 2 4
- - 2 3
- - 4 3
5 - 4 3
5 6 4 3
5 6 4 1
5 6 3 1
Page hit = 7
```

RESULT:

Thus the programs implementing page replacement algorithms are executed successfully.

Ex.No:13.a Date:	FILE ORGANIZATION TECHNIQUE
	SINGLE LEVEL DIRECTORY

AIM:
To write C program to organize the file using single level directory.

ALGORITHM:

Step-1: Start the program.
Step-2: Declare the count, file name, graphical interface.
Step-3: Read the number of files
Step-4: Read the file name
Step-5: Declare the root directory
Step-6: Using the file eclipse function define the files in a single level
Step-7: Display the files
Step-8: Stop the program

FLOWCHART: PROGRAM:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/stat.h> // for mkdir
struct file {
   char name[100];
   int size;
};

int main() {
   int i, count;
   struct file *files;
   printf("Enter number of files: ");
   scanf("%d", &count);
   files = (struct file *)malloc(sizeof(struct file) * count);
   for (i = 0; i < count; i++) {
      printf("Enter file name: ");
      scanf("%s", files[i].name);
      printf("Enter file size: ");
      scanf("%d", &files[i].size);
   }

   // Create a directory to store the files.
   char directory_name[100] = "single_level_directory";
   mkdir(directory_name, 0777);
```

```
    // Store the files in the directory.
    for (i = 0; i < count; i++) {
        char file_path[200] = "";
        strcat(file_path, directory_name);
        strcat(file_path, "/");
        strcat(file_path, files[i].name);
        FILE *fp = fopen(file_path, "wb");
        if (fp == NULL) {
            printf("Error opening file %s\n", file_path);
            exit(1);
        }
        fwrite(files[i].name, sizeof(char), strlen(files[i].name), fp);
        fwrite(&files[i].size, sizeof(int), 1, fp);
        fclose(fp);
    }

    // Display the files in the directory.
    printf("Files in directory %s:\n", directory_name);
    for (i = 0; i < count; i++) {
        char file_path[200] = "";
        strcat(file_path, directory_name);
        strcat(file_path, "/");
        strcat(file_path, files[i].name);
        FILE *fp = fopen(file_path, "rb");
        if (fp == NULL) {
            printf("Error opening file %s\n", file_path);
            exit(1);
        }
        char name[100];
        int size;
        fread(name, sizeof(char), strlen(files[i].name), fp);
        name[strlen(files[i].name)] = '\0'; // Adding null terminator
        fread(&size, sizeof(int), 1, fp);
        printf("%s (%d bytes)\n", name, size);
        fclose(fp);
    }
    free(files); // Free dynamically allocated memory
    return 0;
}
```

OUTPUT:

```
Enter number of files: 2
Enter file name: test1
Enter file size: 2
Enter file name: test2
Enter file size: 34
Files in directory single_level_directory:
test1e_level_directory/test2 (2 bytes)
test2e_level_directory/test2 (34 bytes)


/single_level_directory# ls -l
total 0
-rw-r--r-- 1 root root 9 May 14 22:02 test1
-rw-r--r-- 1 root root 9 May 14 22:02 test2
```

RESULT:

Thus the programs implementing file using Single level directory algorithms are executed successfully.

Ex.No:13.b Date:	FILE ORGANIZATION TECHNIQUE TWO LEVEL DIRECTORY

AIM:
To write C program to organize the file using two level directory.

ALGORITHM:

Step-1: Start the program.
Step-2: Declare the count, file name, graphical interface.
Step-3: Read the number of files
Step-4: Read the file name
Step-5: Declare the root directory
Step-6:Using the file eclipse function define the files in a singlelevel
Step-7: Display the files
Step-8: Stop the program

PROGRAM:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <unistd.h> // Include for access() function
#define MAX_FILENAME_LEN 100
#define MAX_SUBDIRS 10

int main() {
    char filename[MAX_FILENAME_LEN];
    char subdir_name[MAX_FILENAME_LEN];
    char path[MAX_FILENAME_LEN];
    char subdir_paths[MAX_SUBDIRS][MAX_FILENAME_LEN];
    int num_subdirs = 0;

    // Create the top-level directory
    if (mkdir("top_level", 0755) == -1) {
        perror("mkdir");
        exit(1);
    }

    // Prompt the user for a filename
    printf("Enter a filename: ");
    scanf("%s", filename);

    // Check if the filename is too long
    if (strlen(filename) > MAX_FILENAME_LEN) {
        fprintf(stderr, "Error: filename is too long.\n");
        exit(1);
```

```
}

// Prompt the user for a subdirectory name
printf("Enter a subdirectory name: ");
scanf("%s", subdir_name);

// Check if the subdirectory name is too long
if (strlen(subdir_name) > MAX_FILENAME_LEN) {
    fprintf(stderr, "Error: subdirectory name is too long.\n");
    exit(1);
}

// Create the subdirectory if it does not already exist
snprintf(path, sizeof(path), "top_level/%s", subdir_name);
if (mkdir(path, 0755) == -1) {
    perror("mkdir");
}

// Copy the filename to each subdirectory
snprintf(path, sizeof(path), "top_level/%s/%s", subdir_name, filename);
if (access(path, F_OK) != -1) {
    fprintf(stderr, "Error: file already exists.\n");
    exit(1);
}
if (link(filename, path) == -1) {
    perror("link");
    exit(1);
}

// Store the subdirectory paths
snprintf(subdir_paths[num_subdirs++], sizeof(subdir_paths[0]), "top_level/%s", subdir_name);

// Prompt the user for additional subdirectories and repeat the process
while (1) {
    printf("Do you want to add another subdirectory? (y/n) ");
    char answer;
    scanf(" %c", &answer);
    if (answer == 'n') {
        break;
    }
    printf("Enter a subdirectory name: ");
    scanf("%s", subdir_name);
    if (strlen(subdir_name) > MAX_FILENAME_LEN) {
        fprintf(stderr, "Error: subdirectory name is too long.\n");
        exit(1);
    }
    snprintf(path, sizeof(path), "top_level/%s", subdir_name);
    if (mkdir(path, 0755) == -1) {
        perror("mkdir");
    }
```

```
        // Store the subdirectory paths
        snprintf(subdir_paths[num_subdirs++], sizeof(subdir_paths[0]), "top_level/%s", subdir_name);
    }

    // Print out the file and subdirectory paths
    printf("File path: top_level/%s/%s\n", subdir_name, filename);
    printf("Subdirectory paths:\n");
    for (int i = 0; i < num_subdirs; i++) {
        printf("- %s\n", subdir_paths[i]);
    }

    return 0;
}
```

OUTPUT:

```
Enter a filename: myfile.txt
Enter a subdirectory name: mydir
Do you want to add another subdirectory? (y/n) y
Enter a subdirectory name: mydir2
Do you want to add another subdirectory? (y/n) n
File path: top_level/mydir/myfile.txt
Subdirectory paths:
- top_level/mydir
- top_level/mydir2
```

RESULT:

Thus the C programs implementing file organization techniques are executed successfully.

Ex.No:14.a	FILEALLOCATION STRATEGIES
Date:	SEQUENTIAL

AIM:
To write a C program for sequential file for processing the student information.

ALGORITHM:

Step-1: Start the program.
Step-2: Get the number of records user want to store in the system.
Step-3: Using Standard Library function open the file to write the data into the file.
Step-4: Store the entered information in the system.
Step-5: Using do.. While statement and switch case to create the options such as1-DISPLAY, 2.SEARCH, 3.EXIT.
Step-6: Close the file using fclose() function.
Step-7: Process it and display the result.
Step-8: Stop the program.

PROGRAM:
```
#include <stdio.h>
#include <conio.h>

typedef struct {
   int sno;
   char name[25];
   int m1, m2, m3;
} STD;

void display(FILE *);
int search(FILE *);

void main() {
   int i, n, sno_key, opn;
   FILE *fp;
   clrscr();

   printf("How many records?\n");
   scanf("%d", &n);

   fp = fopen("stud.dat", "w");
   for (i = 0; i < n; i++) {
      printf("Enter the student information %d (sno, Name, M1, M2, M3): ", i + 1);
      scanf("%d %s %d %d %d", &sno_key, s.name, &s.m1, &s.m2, &s.m3);
      fwrite(&s, sizeof(s), 1, fp);
   }
   fclose(fp);

   fp = fopen("stud.dat", "r");
```

```
    do {
        printf("1-DISPLAY\n2.SEARCH\n3.EXIT\nYOUR OPTION: ");
        scanf("%d", &opn);
        switch (opn) {
            case 1:
                printf("\nStudent Records in the file\n");
                display(fp);
                break;
            case 2:
                printf("Read sno of the student to be searched: ");
                scanf("%d", &sno_key);
                if (search(fp, sno_key)) {
                    printf("Success!! Record found in the file\n");
                    printf("%d\t%s\t%d\t%d\t%d\n", s.sno, s.name, s.m1, s.m2, s.m3);
                } else {
                    printf("Failure!! Record %d not found\n", sno_key);
                }
                break;
            case 3:
                printf("Exit !! Press any key\n");
                getch();
                break;
            default:
                printf("Invalid option!!! Try again!!\n");
                break;
        }
    } while (opn != 3);
    fclose(fp);
}

void display(FILE *fp) {
    STD s;
    rewind(fp);
    while (fread(&s, sizeof(s), 1, fp)) {
        printf("%d\t%s\t%d\t%d\t%d\n", s.sno, s.name, s.m1, s.m2, s.m3);
    }
}

int search(FILE *fp, int sno_key) {
    STD s;
    rewind(fp);
    while (fread(&s, sizeof(s), 1, fp)) {
        if (s.sno == sno_key)
            return 1;
    }
    return 0;
}
```

OUTPUT:

```
How many records? 3
Enter the student information: 1 (sno,Name,M1,M2,M3): 1001 John 90 85 95
Enter the student information: 2 (sno,Name,M1,M2,M3): 1002 Mary 80 90 85
Enter the student information: 3 (sno,Name,M1,M2,M3): 1003 Tom 95 85 90
1-DISPLAY
2-SEARCH
3-EXIT
YOUR OPTION: 1

Student Records in the file:
1001    John    90      85      95
1002    Mary    80      90      85
1003    Tom     95      85      90

1-DISPLAY
2-SEARCH
3-EXIT
YOUR OPTION: 2
Read sno of the student to be searched: 1004
Failure!! Record 1004 not found

1-DISPLAY
2-SEARCH
3-EXIT
YOUR OPTION: 3

Exit !! press key
```

RESULT:

Thus the C program implementing sequential file for processing the student information is executed successfully.

Ex.No:14.b **Date:**	**FILEALLOCATION STRATEGIES**
	LINKED

AIM:
To write a C program for random access file for processing the employee details.

ALGORITHM:

Step-1: Start the program.
Step-2: Get the number of records user want to store in the system.
Step-3:Using Standard Library function open the file to write the data into the file.
Step-4: Store the entered information in the system.
Step-5: Using do.. While statement and switch case to create the options such
as1-DISPLAY, 2.SEARCH, 3.EXIT.
Step-6: Close the file using fclose() function.
Step-7: Process it and display the result.
Step-8: Stop the program.

PROGRAM:

```
#include <stdio.h>
#include <conio.h>
#include <string.h>

struct record {
    char empname[20];
    int age;
    float salary;
};

typedef struct record person;

void main() {
    person employee;
    int i, n, rec;
    FILE *fp;

    printf("How many records: ");
    scanf("%d", &n);

    fp = fopen("PEOPLE.txt", "w");

    for (i = 0; i < n; i++) {
        printf("Enter the employee information %d (EmpName, Age, Salary): ", i + 1);
        scanf("%s %d %f", employee.empname, &employee.age, &employee.salary);
        fwrite(&employee, sizeof(person), 1, fp);
    }

    fclose(fp);

    FILE *people;
    people = fopen("PEOPLE.txt", "r");

    printf("Which record do you want to read from file? ");
    scanf("%d", &rec);

    while (rec >= 0) {
        fseek(people, rec * sizeof(person), SEEK_SET);
        int result = fread(&employee, sizeof(person), 1, people);

        if (result == 1) {
            printf("\n RECORD %d\n", rec);
            printf("Given name: %s\n", employee.empname);
            printf("Age: %d years\n", employee.age);
            printf("Current salary: $ %.2f\n\n", employee.salary);
        } else {
            printf("\n RECORD %d not found!\n\n", rec);
        }

        printf("Which record do you want (0 to %d): ", n - 1);
        scanf("%d", &rec);
    }

    fclose(people);
    getch();
}
```

OUTPUT:

```
How many records: 2
Enter the employee information: 1(EmpName, Age, Salary): John 25 50000
Enter the employee information: 2(EmpName, Age, Salary): Sarah 30 70000
Which record do you want to read from file (0 to 2)? 1

RECORD 1
Given name: Sarah
Age: 30 years
Current salary: $  70000.00

Which record do you want to read (0 to 2)? 0

RECORD 0
Given name: John
Age: 25 years
Current salary: $  50000.00
```

RESULT:

Thus the programs implementing random access file for processing the employee details are executed successfully.

Ex.No:14.c Date:	FILEALLOCATION STRATEGIES INDEXED

AIM:
To write a C program for random access file for processing the employee details.

ALGORITHM:

Step-1: Start the program.
Step-2: Get the number of records user want to store in the system.
Step-3: Using Standard Library function open the file to write the data into the file.
Step-4: Store the entered information in the system.
Step-5: Using do.. While statement and switch case to create the options such as1-DISPLAY, 2.SEARCH, 3.EXIT.
Step-6: Close the file using fclose() function.
Step-7: Process it and display the result.
Step-8: Stop the program.

PROGRAM:
```
#include <stdio.h>
#include <stdlib.h>

int main() {
    int f[50] = {0}; // Initialize array with zeros
    int index[50], n, ind, i, j, k, count, c;

    do {
        printf("Enter the index block: ");
        scanf("%d", &ind);

        if (f[ind] != 1) {
            printf("Enter no of blocks needed and no of files for the index %d on the disk: \n", ind);
            scanf("%d", &n);
        } else {
            printf("%d index is already allocated.\n", ind);
            continue;
        }

        count = 0;
        for (i = 0; i < n; i++) {
            printf("Enter block %d: ", i + 1);
            scanf("%d", &index[i]);
            if (f[index[i]] == 0)
                count++;
        }
        if (count == n) {
            for (j = 0; j < n; j++)
                f[index[j]] = 1;

            printf("Allocated.\n");
            printf("File Indexed.\n");

            for (k = 0; k < n; k++)
                printf("%d-------->%d : %d\n", ind, index[k], f[index[k]]);
        } else {
            printf("File in the index is already allocated.\n");
            printf("Enter another file indexed.\n");
            continue;
        }

        printf("Do you want to enter more files? (Yes - 1/No - 0): ");
        scanf("%d", &c);

    } while (c == 1);

    printf("Exiting program.\n");
    return 0;
}
```

OUTPUT:

```
Enter the index block: 5
Enter no of blocks needed and no of files for the index 5 on the disk :
4
Enter block 1: 1
Enter block 2: 2
Enter block 3: 3
Enter block 4: 5
Allocated
File Indexed
5------- >1 : 1
5-------->2 : 1
5-------->3 : 1
5-------->4 : 1
Do you want to enter more file(Yes - 1/No - 0)1
Enter the index block: 4
4 index is already allocated
Enter the index block: 6
Enter no of blocks needed and no of files for the index 6 on the disk :
2
7 8
A5llocated
File Indexed
6------- >7 : 1
6-------->8 : 1
Do you want to enter more file(Yes - 1/No - 0)0
Exiting program.
```

RESULT:

Thus the C programs implementing file allocation strategies are executed successfully.

Ex.No:15.a **Date:**	**DISK SCHEDULING ALGORITHMS**
	FCFS

AIM:
To write a C program for FCFS disk scheduling algorithms.

ALGORITHM:

Step-1: Start the program.
Step-2: Get the number of records user want to store in the system.
Step-3: Using Standard Library function open the file to write the data into the file.
Step-4: Store the entered information in the system.
Step-5: Using do.. While statement and switch case to create the options such as1-DISPLAY, 2.SEARCH, 3.EXIT.
Step-6: Close the file using fclose() function.
Step-7: Process it and display the result.
Step-8: Stop the program.

PROGRAM:
```
#include<stdio.h>
#include<stdlib.h>

int main() {
   int RQ[100], i, n, TotalHeadMoment = 0, initial;

   printf("Enter the number of Requests\n");
   scanf("%d", &n);

   printf("Enter the Requests sequence\n");
   for (i = 0; i < n; i++)
      scanf("%d", &RQ[i]);

   printf("Enter initial head position\n");
   scanf("%d", &initial);

   // Logic for FCFS disk scheduling
   for (i = 0; i < n; i++) {
      TotalHeadMoment += abs(RQ[i] - initial);
      initial = RQ[i];
   }
   printf("Total head moment is %d\n", TotalHeadMoment);
   return 0;
}
```
OUTPUT:
```
Enter the number of Requests 8
Enter the Requests sequence 55 75 45 88 12 95 56 78
Enter initial head position 50
Total head moment is 318
```

Ex.No:15.b Date:	DISK SCHEDULING ALGORITHMS SSTF

AIM:

To write a C program for SSTF disk scheduling algorithms.

ALGORITHM:

Step-1: Start the program.
Step-2: Get the number of records user want to store in the system.
Step-3: Using Standard Library function open the file to write the data into the file.
Step-4: Store theentered information in the system.
Step-5: Using do..While statement and switchcase to create the options such as
 1- DISPLAY, 2.SEARCH, 3.EXIT.
Step-6: Close the file using fclose() function
Step-7: Process it and display the result.
Step-8: Stop the program.

PROGRAM:

```
#include <stdio.h>
#include <stdlib.h>

int main() {
int RQ[100], I, n, TotalHeadMovement = 0, initial, count = 0;

printf("Enter the number of Requests\n");
scanf("%d", &n);

printf("Enter the Requests sequence\n");
for (i = 0; i < n; i++)
   scanf("%d", &RQ[i]);

printf("Enter initial head position\n");
scanf("%d", &initial);

// Logic for SSTF disk scheduling
while (count != n) {
int min = 1000, d, index;
for (i= 0; i< n; i++) {
   d = abs(RQ[i] – initial);
   if (d < min) {
      min = d;
      index = i;
   }
}
TotalHeadMovement += min;
initial = RQ[index];
RQ[index] = 1000; // Mark as visited
Count++;
}
```

```
    printf(“Total head movement is %d\n”, TotalHeadMovement);
    return 0;
}
```

OUTPUT:

```
Enter the number of Requests 8
Enter the Requests sequence 95 180 34 119 11 123 62 64
Enter initial head position 50
Total head movement is 236
```

Ex.No:15.c Date:	DISK SCHEDULING ALGORITHMS
	SCAN

AIM:
To write a C program for SCAN disk scheduling algorithms.

ALGORITHM:

Step-1: Start the program.
Step-2: Get the number of records user want to store in the system.
Step-3: Using Standard Library function open the file to write the data into the file.
Step-4: Store theentered information in the system.
Step-5: Using do.. While statement and switch case to create the options such
as1- DISPLAY, 2.SEARCH, 3.EXIT.
Step-6: Close the file using fclose() function.
Step-7: Process it and display the result.
Step-8: Stop the program.

PROGRAM:
```
#include <stdio.h>
#include <stdlib.h>

int main() {
   int RQ[100], i, j, n, TotalHeadMoment = 0, initial, size, move;
   printf("Enter the number of Requests\n");
   scanf("%d", &n);
   printf("Enter the Requests sequence\n");
   for (i = 0; i < n; i++)
      scanf("%d", &RQ[i]);
   printf("Enter initial head position\n");
   scanf("%d", &initial);
   printf("Enter total disk size\n");
   scanf("%d", &size);
   printf("Enter the head movement direction for high 1 and for low 0\n");
   scanf("%d", &move);
   // Logic for Scan disk scheduling
   // Sorting the request array
   for (i = 0; i < n; i++) {
      for (j = 0; j < n - i - 1; j++) {
         if (RQ[j] > RQ[j + 1]) {
            int temp = RQ[j];
            RQ[j] = RQ[j + 1];
            RQ[j + 1] = temp;
         }
      }
   }
   // Finding the index where initial head position lies
   int index;
```

```
    for (i = 0; i < n; i++) {
        if (initial < RQ[i]) {
            index = i;
            break;
        }
    }
    // If movement is towards high value
    if (move == 1) {
        for (i = index; i < n; i++) {
            TotalHeadMoment += abs(RQ[i] - initial);
            initial = RQ[i];
        }
        // Last movement for max size
        TotalHeadMoment += abs(size - RQ[i - 1] - 1);
        initial = size - 1;
        for (i = index - 1; i >= 0; i--) {
            TotalHeadMoment += abs(RQ[i] - initial);
            initial = RQ[i];
        }
    } else { // If movement is towards low value
        for (i = index - 1; i >= 0; i--) {
            TotalHeadMoment += abs(RQ[i] - initial);
            initial = RQ[i];
        }
        // Last movement for min size
        TotalHeadMoment += abs(RQ[i + 1] - 0);
        initial = 0;
        for (i = index; i < n; i++) {
            TotalHeadMoment += abs(RQ[i] - initial);
            initial = RQ[i];
        }
    }
    printf("Total head movement is %d\n", TotalHeadMoment);
    return 0;
}
```

OUTPUT:

```
 Enter the number of Requests
 8
 Enter the Requests sequence
 95 180 34 119 11 123 62 64
 Enter initial head position
50
Enter total disk size
200
Enter the head movement direction for high 1 and for low 0
1
Total head movement is 337
```

Ex.No:15.d	DISK SCHEDULING ALGORITHMS
Date:	C-SCAN

AIM:

To write a C program for C-SCAN disk scheduling algorithms.

ALGORITHM:

Step-1: Start the program.
Step-2: Get the number of records user want to store in the system.
Step-3: Using Standard Library function open the file to write the data into the file.
Step-4: Store theentered information in the system.
Step-5: Using do.. While statement and switch case to create the options such as1- DISPLAY, 2.SEARCH, 3.EXIT.
Step-6: Close the file using fclose() function
Step-7: Process it and display the result.
Step-8: Stop the program.

PROGRAM:

```
#include <stdio.h>
#include <stdlib.h>

int main() {
   int RQ[100], i, j, n, TotalHeadMoment = 0, initial, size, move;

   printf("Enter the number of Requests: ");
   scanf("%d", &n);

   printf("Enter the Requests sequence:\n");
   for (i = 0; i < n; i++)
      scanf("%d", &RQ[i]);

   printf("Enter initial head position: ");
   scanf("%d", &initial);

   printf("Enter total disk size: ");
   scanf("%d", &size);

   printf("Enter the head movement direction for high (1) and for low (0): ");
   scanf("%d", &move);

   // Logic for sorting the request array
   for (i = 0; i < n; i++) {
      for (j = 0; j < n - i - 1; j++) {
         if (RQ[j] > RQ[j + 1]) {
            int temp = RQ[j];
            RQ[j] = RQ[j + 1];
            RQ[j + 1] = temp;
            }
        }
    }

    int index;
    for (i = 0; i < n; i++) {
        if (initial < RQ[i]) {
            index = i;
            break;
        }
    }
    // If movement is towards high value
    if (move == 1) {
        for (i = index; i < n; i++) {
            TotalHeadMoment += abs(RQ[i] - initial);
            initial = RQ[i];
        }
        // Last movement for max size
        TotalHeadMoment += abs(size - RQ[i - 1] - 1);
        // Movement max to min disk
        TotalHeadMoment += abs(size - 1 - 0);
        initial = 0;
        for (i = 0; i < index; i++) {
            TotalHeadMoment += abs(RQ[i] - initial);
            initial = RQ[i];
        }
    } else { // If movement is towards low value
        for (i = index - 1; i >= 0; i--) {
            TotalHeadMoment += abs(RQ[i] - initial);
            initial = RQ[i];
        }
        // Last movement for min size
        TotalHeadMoment += abs(RQ[i + 1] - 0);
        // Movement min to max disk
        TotalHeadMoment += abs(size - 1 - 0);
        initial = size - 1;
        for (i = n - 1; i >= index; i--) {
            TotalHeadMoment += abs(RQ[i] - initial);
            initial = RQ[i];
        }
    }
    printf("Total head movement is %d\n", TotalHeadMoment);
    return 0;
}
```

OUTPUT:

```
Enter the number of Requests 8
Enter the Requests sequence 95 180 34 119 11 123 62 64
Enter initial head position 50
Enter total disk size 200
Enter the head movement direction for high (1) and for low (0): 1
Total head movement is 382
```

Ex.No:15.e **Date:**	**DISK SCHEDULING ALGORITHMS**
	LOOK

AIM:

To write a C program for LOOK disk scheduling algorithms.

ALGORITHM:

Step-1: Start the program.
Step-2: Get the number of records user want to store in the system.
Step-3: Using Standard Library function open the file to write the data into the file.
Step-4: Store theentered information in the system.
Step-5: Using do.. While statement and switch case to create the options such as1- DISPLAY, 2.SEARCH, 3.EXIT.
Step-6: Close the file using fclose() function
Step-7: Process it and display the result.
Step-8: Stop the program.

PROGRAM:

```
#include <stdio.h>
#include <stdlib.h>

int main() {
    int RQ[100], i, j, n, TotalHeadMoment = 0, initial, size, move;

    printf("Enter the number of Requests: ");
    scanf("%d", &n);

    printf("Enter the Requests sequence: ");
    for (i = 0; i < n; i++)
        scanf("%d", &RQ[i]);

    printf("Enter initial head position: ");
    scanf("%d", &initial);

    printf("Enter total disk size: ");
    scanf("%d", &size);

    printf("Enter the head movement direction for high 1 and for low 0: ");
    scanf("%d", &move);

    // Logic for sorting the request array
    for (i = 0; i < n; i++) {
        for (j = 0; j < n - i - 1; j++) {
            if (RQ[j] > RQ[j + 1]) {
                int temp = RQ[j];
                RQ[j] = RQ[j + 1];
                RQ[j + 1] = temp;
            }
        }
    }

    int index;
    for (i = 0; i < n; i++) {
        if (initial < RQ[i]) {
            index = i;
            break;
        }
    }

    // If movement is towards high value
    if (move == 1) {
        for (i = index; i < n; i++) {
            TotalHeadMoment += abs(RQ[i] - initial);
            initial = RQ[i];
        }
        for (i = index - 1; i >= 0; i--) {
            TotalHeadMoment += abs(RQ[i] - initial);
            initial = RQ[i];
        }
    } else { // If movement is towards low value
        for (i = index - 1; i >= 0; i--) {
            TotalHeadMoment += abs(RQ[i] - initial);
            initial = RQ[i];
        }
        for (i = index; i < n; i++) {
            TotalHeadMoment += abs(RQ[i] - initial);
            initial = RQ[i];
        }
    }

    printf("Total head movement is %d\n", TotalHeadMoment);
    return 0;
}
```

OUTPUT:

```
Enter the number of Requests: 8
Enter the Requests sequence: 95 180 34 119 11 123 62 64
Enter initial head position: 50
Enter total disk size: 200
Enter the head movement direction for high 1 and for low 0: 1
Total head movement is 299
```

Ex.No:15.f **Date:**	**DISK SCHEDULING ALGORITHMS**
	C-LOOK

AIM:
To write a C program for C-LOOK disk scheduling algorithms.

ALGORITHM:

Step-1: Start the program.
Step-2: Get the number of records user want to store in the system.
Step-3: Using Standard Library function open the file to write the data into the file.
Step-4: Store the entered information in the system.
Step-5: Using do..While statement and switch case to create the options such as 1-DISPLAY, 2.SEARCH, 3.EXIT.
Step-6: Close the file using fclose() function.
Step-7: Process it and display the result.
Step-8: Stop the program.

PROGRAM:

```
#include<stdio.h>
#include<stdlib.h>

int main() {
   int RQ[100], i, j, n, TotalHeadMoment = 0, initial, size, move;
   printf("Enter the number of Requests\n");
   scanf("%d", &n);
   printf("Enter the Requests sequence\n");
   for (i = 0; i < n; i++) {
      scanf("%d", &RQ[i]);
   }
   printf("Enter initial head position\n");
   scanf("%d", &initial);
   printf("Enter total disk size\n");
   scanf("%d", &size);
   printf("Enter the head movement direction for high 1 and for low 0\n");
   scanf("%d", &move);
   // Logic for C-look disk scheduling
   // Logic for sorting the request array
   for (i = 0; i < n; i++) {
      for (j = 0; j < n - i - 1; j++) {
         if (RQ[j] > RQ[j + 1]) {


            int temp;
            temp = RQ[j];
            RQ[j] = RQ[j + 1];
            RQ[j + 1] = temp;
         }
      }
```

```
    }
    int index;
    for (i = 0; i < n; i++) {
        if (initial < RQ[i]) {
            index = i;
            break;
        }
    }
    // If movement is towards high value
    if (move == 1) {
        for (i = index; i < n; i++) {
            TotalHeadMoment += abs(RQ[i] - initial);
            initial = RQ[i];
        }
        for (i = 0; i < index; i++) {
            TotalHeadMoment += abs(RQ[i] - initial);
            initial = RQ[i];
        }
    }
    // If movement is towards low value
    else {
        for (i = index - 1; i >= 0; i--) {
            TotalHeadMoment += abs(RQ[i] - initial);
            initial = RQ[i];
        }
        for (i = n - 1; i >= index; i--) {
            TotalHeadMoment += abs(RQ[i] - initial);
            initial = RQ[i];
        }
    }
    printf("Total head movement is %d\n", TotalHeadMoment);
    return 0;
}
```

OUTPUT:

```
Enter the number of Requests 8
Enter the Requests sequence 95180 34 119 11 123 62 64
Enter initial head position 50
Enter total disk size 200
 Enter the head movement direction for high 1 and for low 0 1
Total head movement is 322
```

RESULT

Thus the C program implementing disk scheduling algorithms are executed successfully.

Ex.No:16 Date:	INSTALLATION OF GUEST OS USING VMWARE

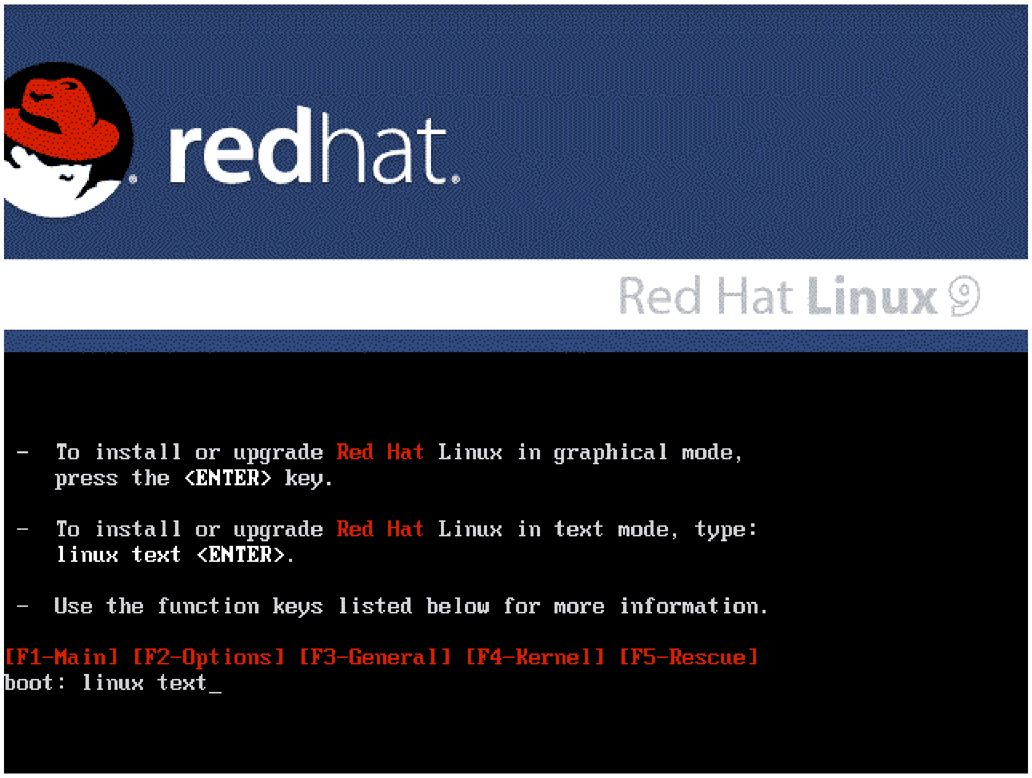
AIM:
To install any guest operating system like Linux using VMware

PROCEDURE:

Linux Installation

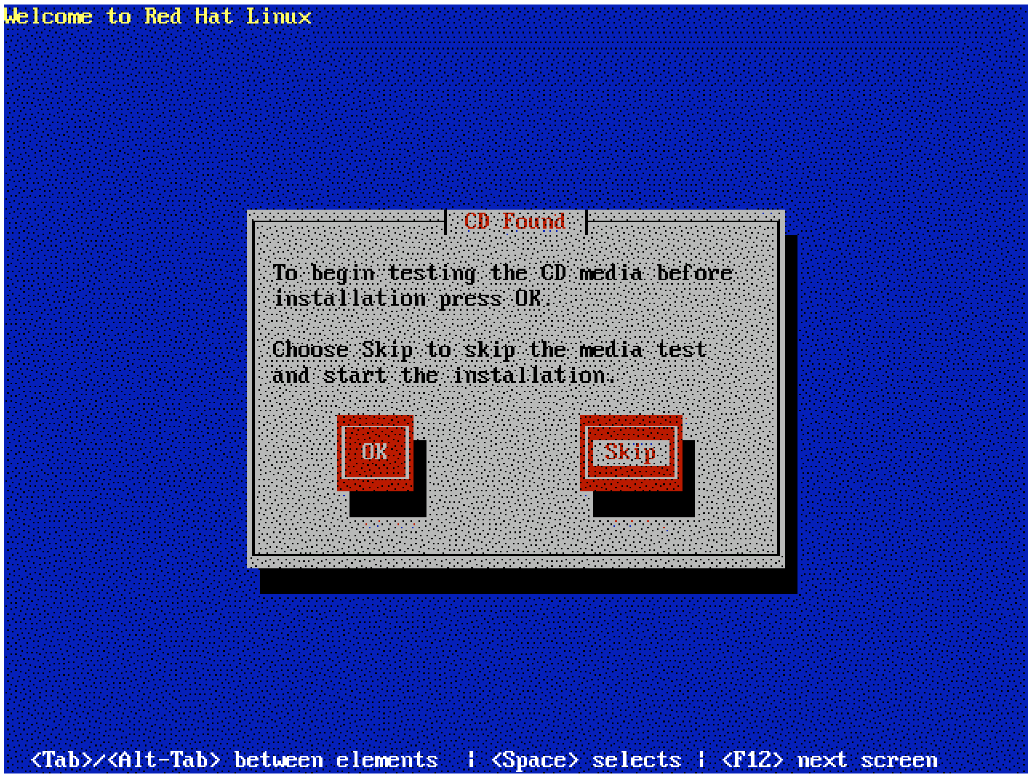
Now you are going to install Red Hat 9.0. The first step is to insert the Red Hat 9.0 Disc 1 into your CD-ROM drive. Next, power on the system. The system boots off of the CD- ROM and begins the Red Hat installation program. Follow these steps to complete the installation of RedHat:

1. When the **Red Hat Installation** screen appears (the first screen) type **linux text** at the **boot:** prompt and press **Enter,** as shown in the followingscreen.



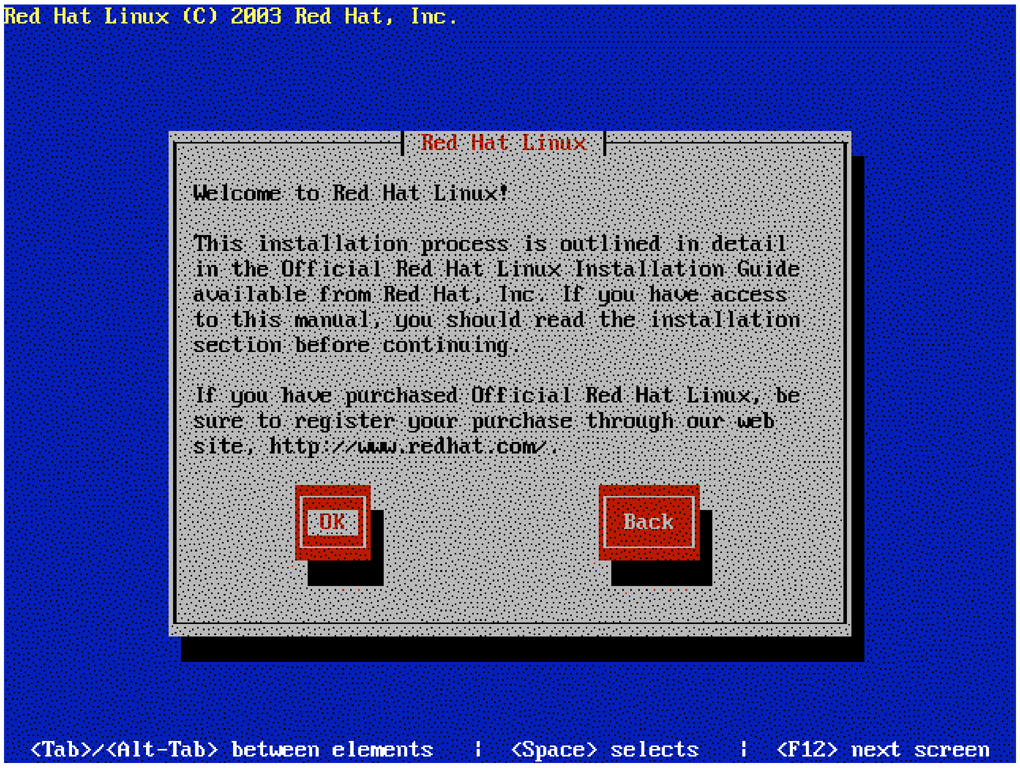
```
-  To install or upgrade Red Hat Linux in graphical mode,
   press the <ENTER> key.

-  To install or upgrade Red Hat Linux in text mode, type:
   linux text <ENTER>.

-  Use the function keys listed below for more information.

[F1-Main] [F2-Options] [F3-General] [F4-Kernel] [F5-Rescue]
boot: linux text_
```

2. Press the **Tab** key until **Skip** is highlighted, and then press **Enter**.



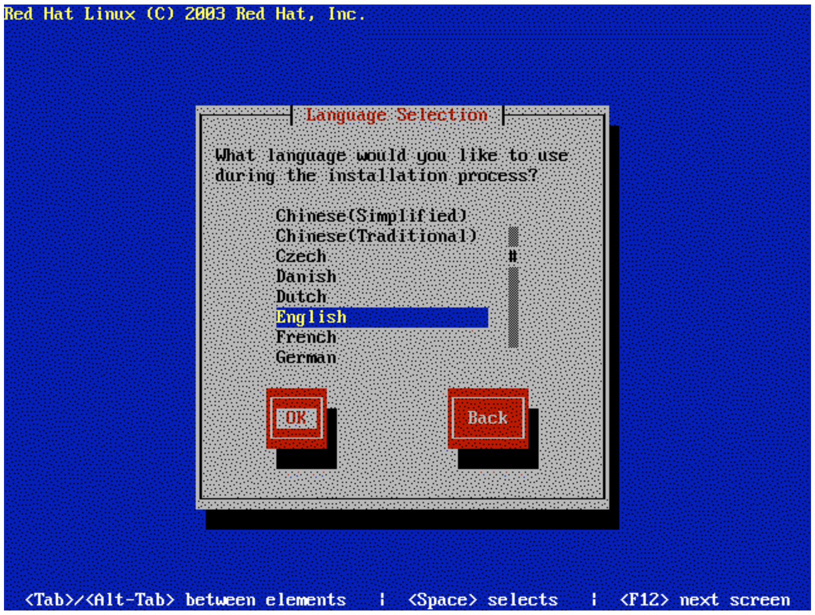
```
Welcome to Red Hat Linux

CD Found

To begin testing the CD media before
installation press OK.

Choose Skip to skip the media test
and start the installation.

 <Tab>/<Alt-Tab> between elements  |  <Space> selects  |  <F12> next screen
```

3. The “Welcome’ screen appears. Press **Enter**.



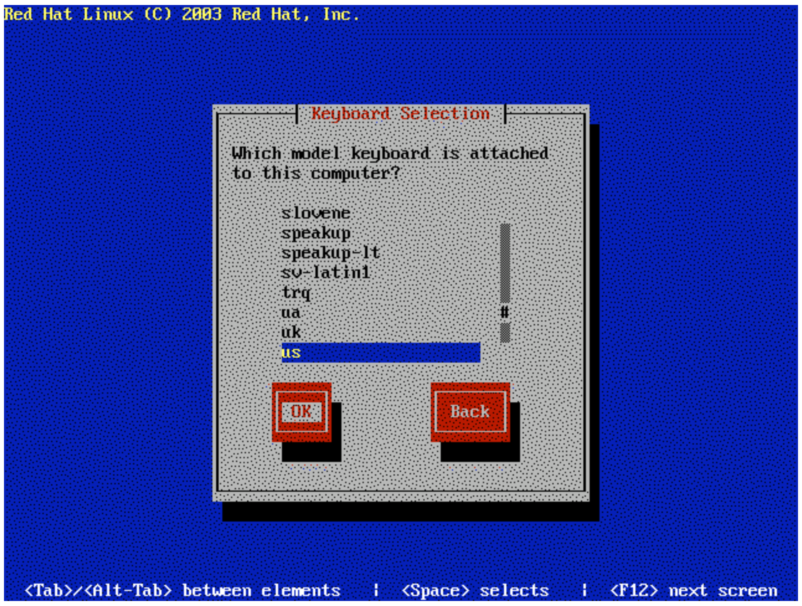
```
Red Hat Linux (C) 2003 Red Hat, Inc.

Red Hat Linux

Welcome to Red Hat Linux!

This installation process is outlined in detail
in the Official Red Hat Linux Installation Guide
available from Red Hat, Inc. If you have access
to this manual, you should read the installation
section before continuing.

If you have purchased Official Red Hat Linux, be
sure to register your purchase through our web
site, http://www.redhat.com/.

 <Tab>/<Alt-Tab> between elements  |  <Space> selects  |  <F12> next screen
```

4. The ‘Language Selection’ screen appears. Ensure that the language is setto **English**. Press the **Tab** key until **OK** is highlighted, and then press **Enter**.



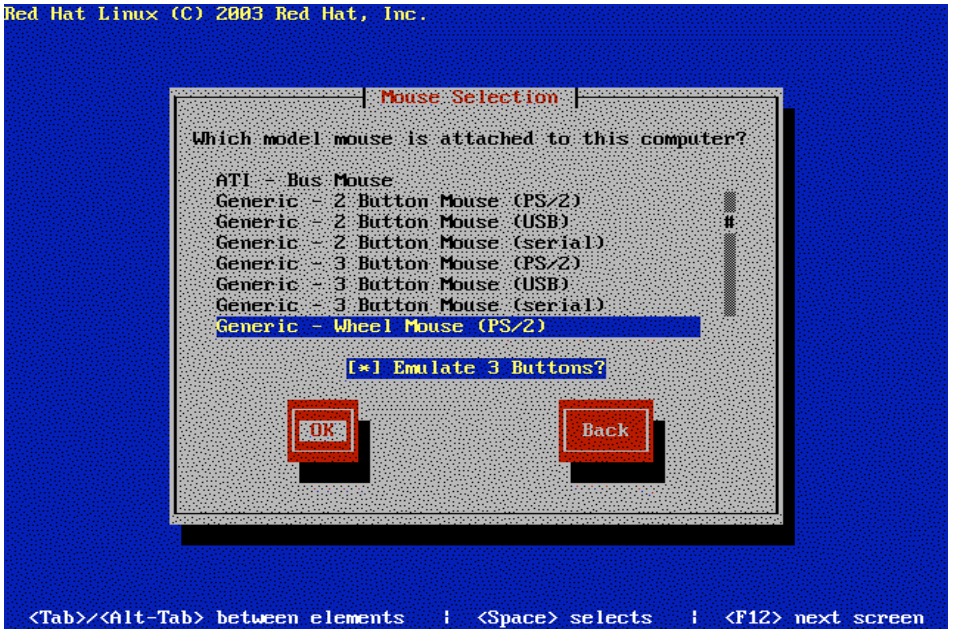
```
Red Hat Linux (C) 2003 Red Hat, Inc.

Language Selection

What language would you like to use
during the installation process?

Chinese(Simplified)
Chinese(Traditional)
Czech
Danish
Dutch
English
French
German

OK          Back

<Tab>/<Alt-Tab> between elements  |  <Space> selects  |  <F12> next screen
```

1. The ‘Keyboard Selection’ screen appears. Accept the default keyboard **us.**
Press the **Tab** key until **OK** is highlighted and press **Enter**.



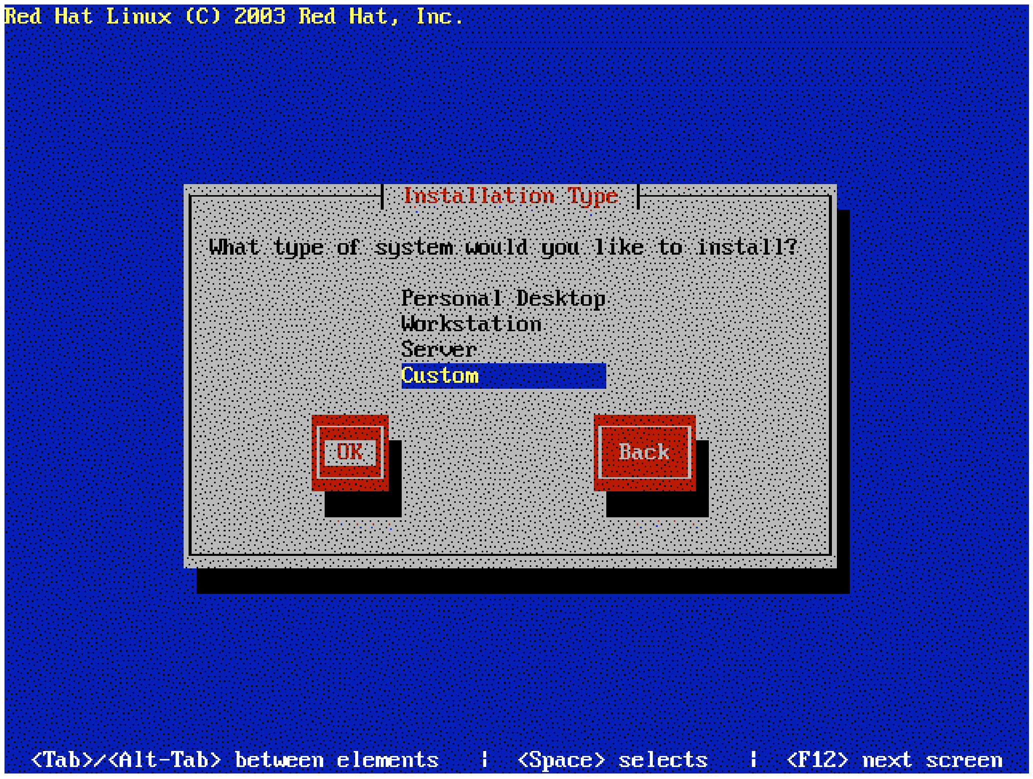
```
Red Hat Linux (C) 2003 Red Hat, Inc.

Keyboard Selection

Which model keyboard is attached
to this computer?

slovene
speakup
speakup-lt
sv-latin1
trq
ua
uk
us

OK          Back

<Tab>/<Alt-Tab> between elements  |  <Space> selects  |  <F12> next screen
```

2. The ‘Mouse Selection’ screen appears. Press the **Tab** key until the boxnext to **Emulate 3 Buttons** is selected and press **Space Bar** to place anasterisk in the brackets **[*].** Next, press the **Tab** key until **OK** is highlighted and then press **Enter**.



```
Red Hat Linux (C) 2003 Red Hat, Inc.

Mouse Selection

Which model mouse is attached to this computer?

  ATI - Bus Mouse
  Generic - 2 Button Mouse (PS/2)
  Generic - 2 Button Mouse (USB)
  Generic - 2 Button Mouse (serial)
  Generic - 3 Button Mouse (PS/2)
  Generic - 3 Button Mouse (USB)
  Generic - 3 Button Mouse (serial)
  Generic - Wheel Mouse (PS/2)

[*] Emulate 3 Buttons?

OK          Back

<Tab>/<Alt-Tab> between elements  |  <Space> selects  |  <F12> next screen
```

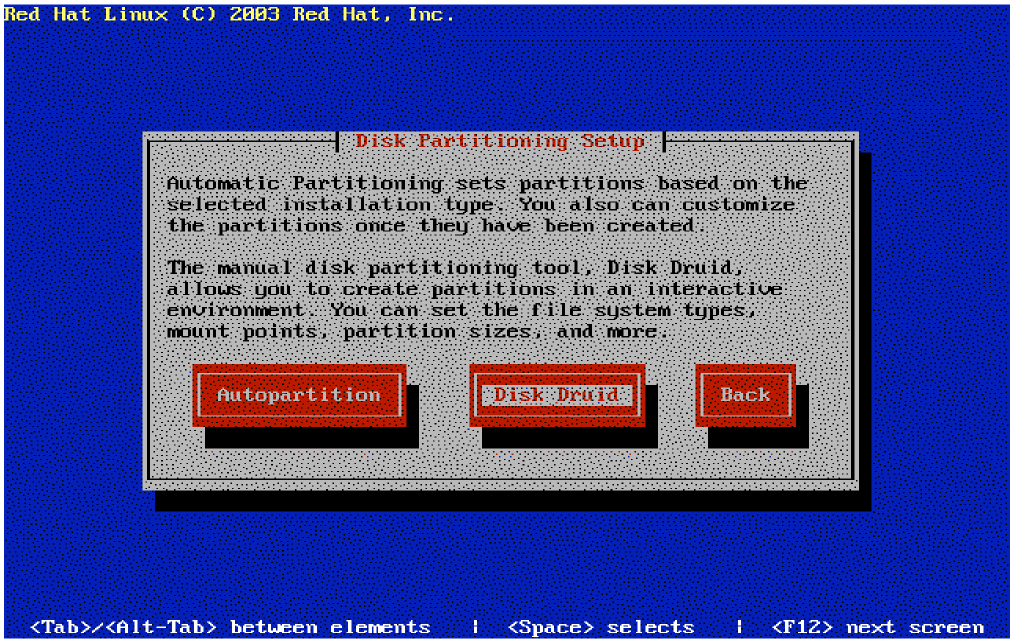
3. The 'Installation Type' screen appears. Use the arrow keys to highlight **Custom**. Press the **Tab** key until **OK** is highlighted and then press**Enter**.



```
Red Hat Linux (C) 2003 Red Hat, Inc.

Installation Type

What type of system would you like to install?

        Personal Desktop
        Workstation
        Server
        Custom

OK          Back

<Tab>/<Alt-Tab> between elements  |  <Space> selects  |  <F12> next screen
```

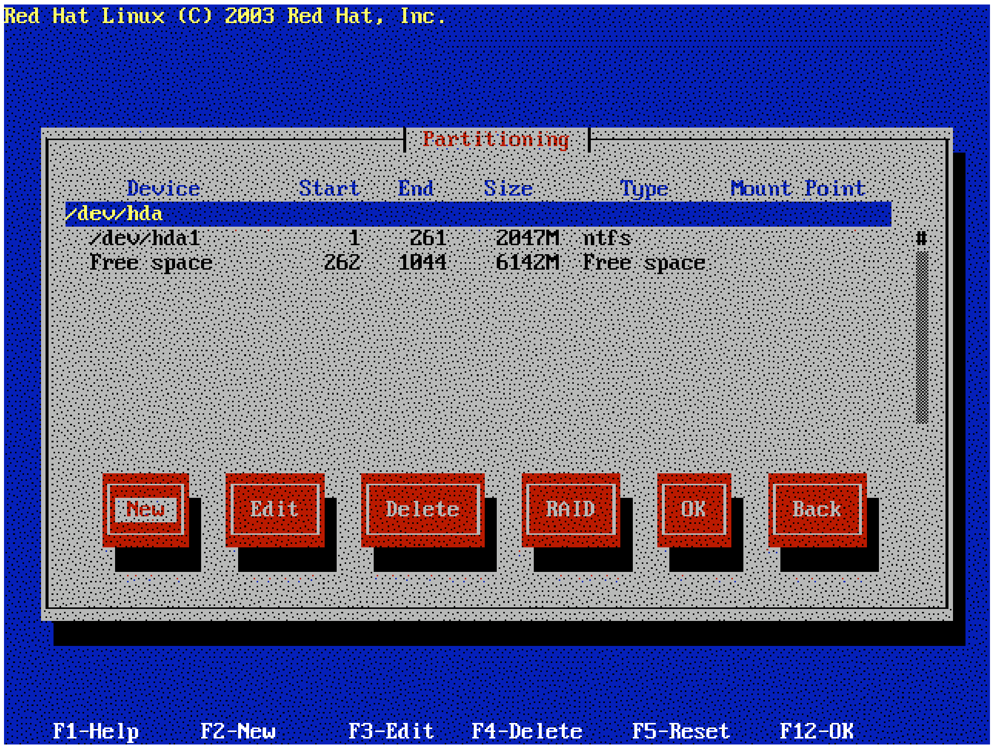
4. The ‘Disk Partitioning Setup’ screen appears. Press the **Tab** key until **Disk Druid** is highlighted, and then press **Enter**.

Note that the values used to partition the hard drive may need to be altered based onthe memory and hard drive size of the system that you are using.



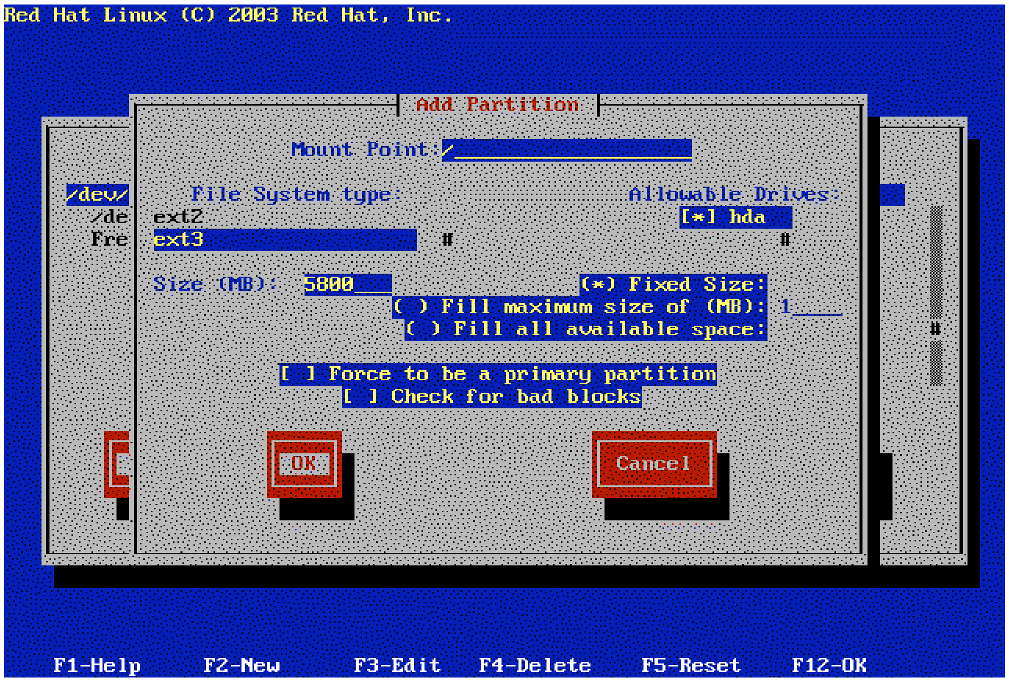
```
Red Hat Linux (C) 2003 Red Hat, Inc.

Disk Partitioning Setup

Automatic Partitioning sets partitions based on the
selected installation type. You also can customize
the partitions once they have been created.

The manual disk partitioning tool, Disk Druid,
allows you to create partitions in an interactive
environment. You can set the file system types,
mount points, partition sizes, and more.

<Tab>/<Alt-Tab> between elements   |  <Space> selects   |  <F12> next screen
```

5. The ‘Partitioning’ screen appears. Press the **Tab** key until **New** ishighlighted, and the press **Enter**.



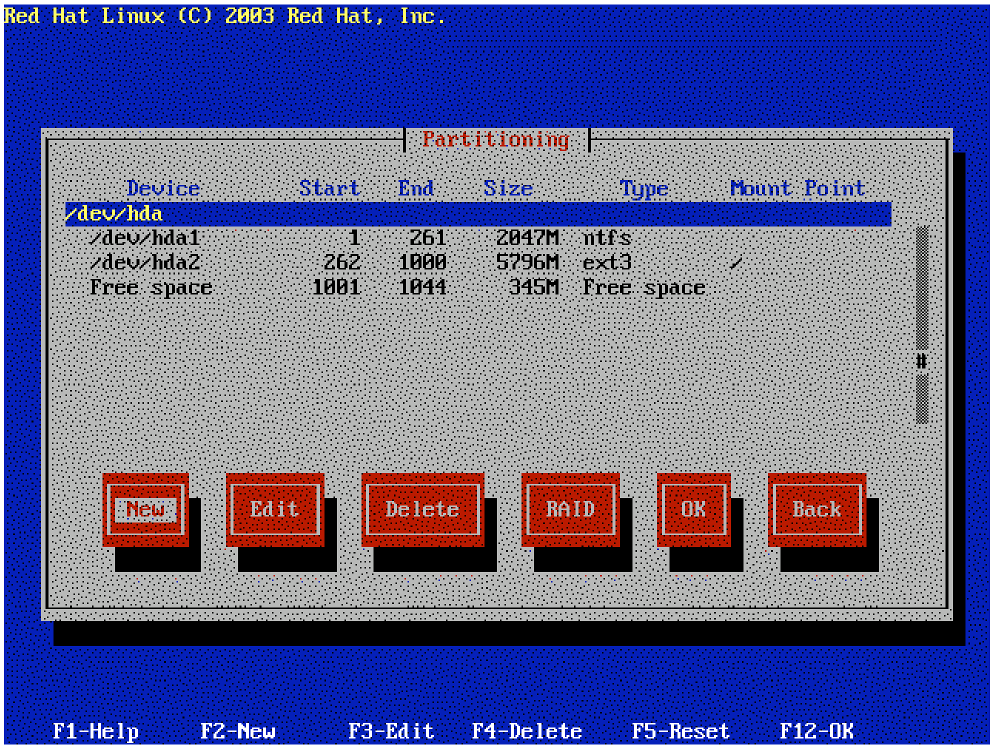
```
Red Hat Linux (C) 2003 Red Hat, Inc.

Partitioning

     Device          Start    End     Size        Type      Mount Point
/dev/hda
  /dev/hda1              1     261    2047M  ntfs
  Free space           262    1044    6142M  Free space

    F1-Help     F2-New     F3-Edit    F4-Delete    F5-Reset    F12-OK
```

6. The ‘Add Partition’ screen appears. In Mount Point: type **/**. Press the **Tab**key until the cursor is in the **Size (MB)**: field. Enter **5800**. Press the **Tab** key until **OK** is highlighted, and then press **Enter**.



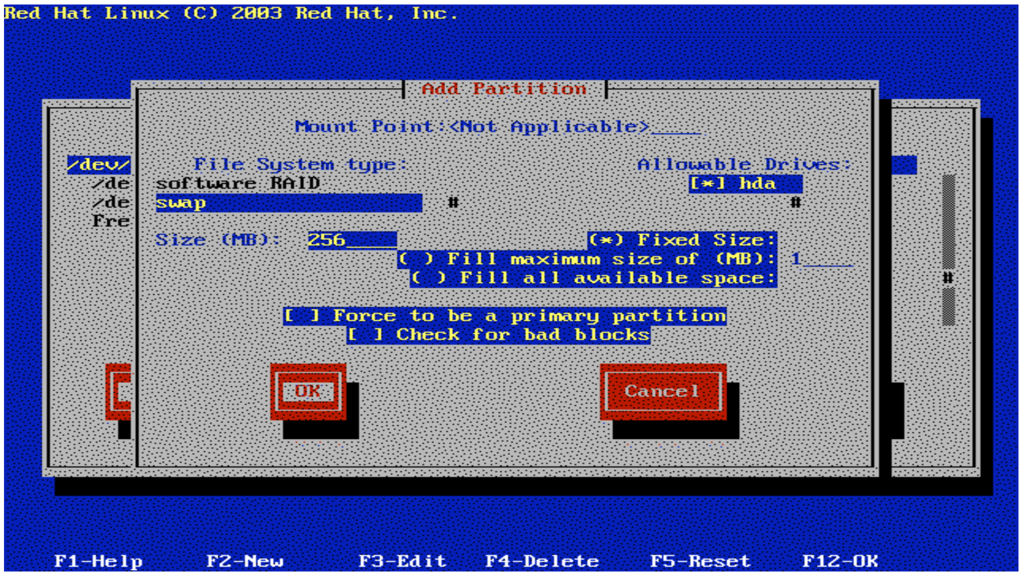
```
Red Hat Linux (C) 2003 Red Hat, Inc.

                         Add Partition

                 Mount Point: /_______________

/dev/      File System type:              Allowable Drives:
 /de    ext2                                  [*] hda
 Fre    ext3                        #                  #

        Size (MB):   5800___          (*) Fixed Size:
                                ( ) Fill maximum size of (MB): 1____
                                 ( ) Fill all available space:

                   [ ] Force to be a primary partition
                        [ ] Check for bad blocks

                  OK                      Cancel

    F1-Help     F2-New      F3-Edit    F4-Delete    F5-Reset     F12-OK
```

7. The ‘Partitioning’ screen reappears. With the arrow and **Tab** keys, highlight **Free Space**, as shown in the following screen. Afterwards, pressthe **Tab** key until **New** is highlighted, and then press **Enter**.



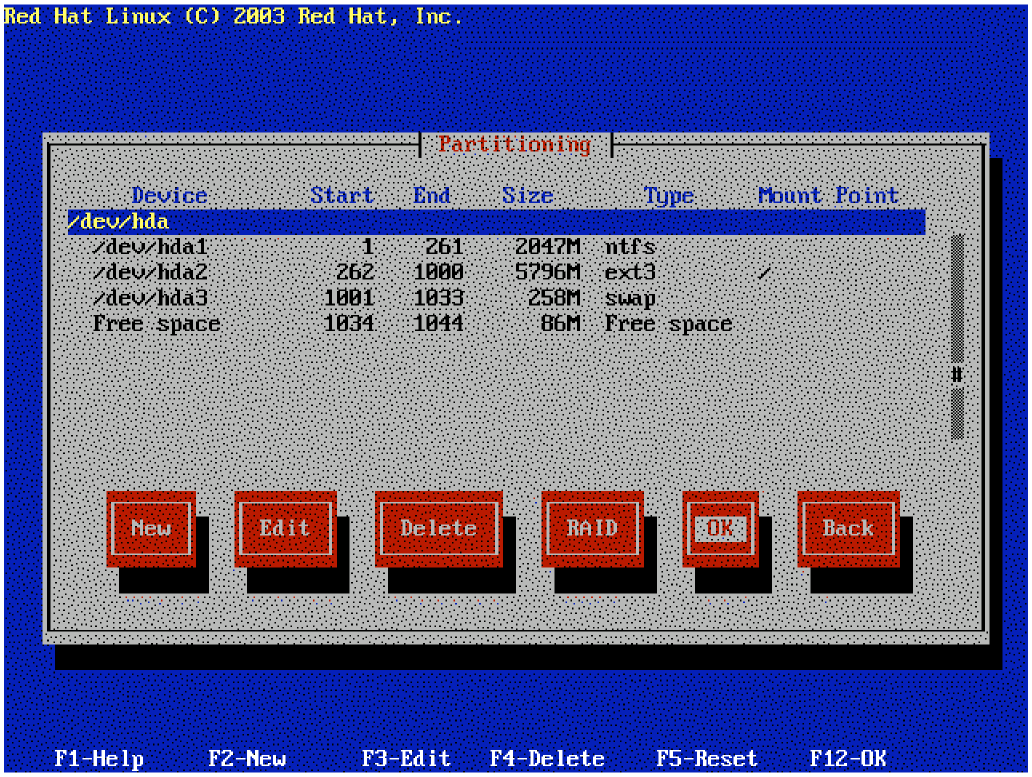
```
Red Hat Linux (C) 2003 Red Hat, Inc.

                          Partitioning

       Device          Start    End    Size        Type      Mount Point
/dev/hda
  /dev/hda1                1    261    2047M  ntfs
  /dev/hda2              262   1000    5796M  ext3             /
  Free space            1001   1044     345M  Free space

     New      Edit      Delete      RAID      OK      Back

    F1-Help     F2-New      F3-Edit    F4-Delete    F5-Reset     F12-OK
```

8. The ‘Add Partition’ screen appears. Press the **Tab** key once to select the**File System type**: field. Using the arrow keys, highlight **swap.** Press the**Tab** key until the **Size (MB)**: field is selected. Enter **256**. Press the **Tab** key until **OK** is highlighted, and then press **Enter**. These fields and selections are shown in the following screen.



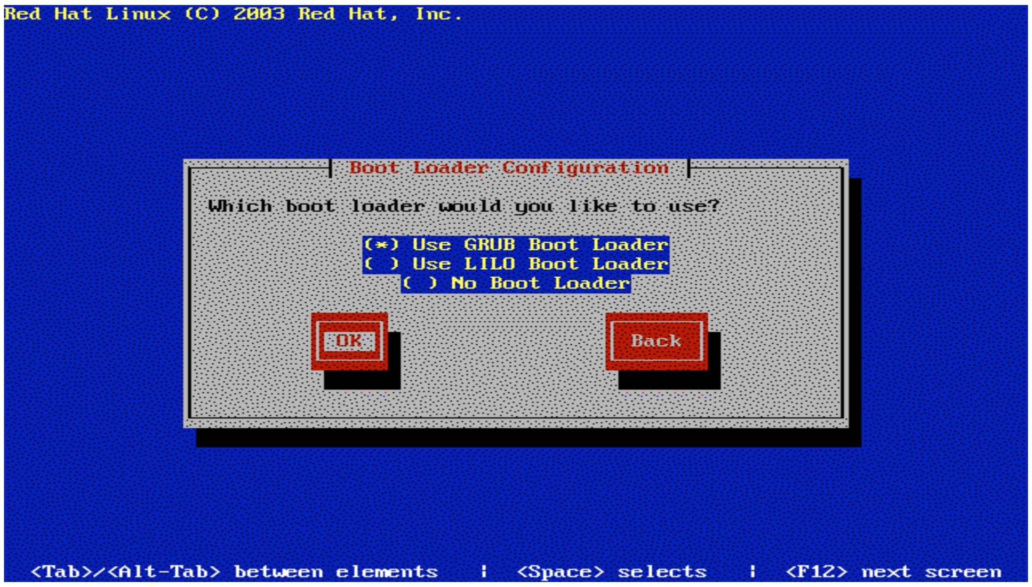
```
Red Hat Linux (C) 2003 Red Hat, Inc.

                          Add Partition

            Mount Point:<Not Applicable>____

/dev/     File System type:                  Allowable Drives:
 /de    software RAID                          [*] hda
 /de    swap                        #                      #
 Fre
        Size (MB):  256____              (*) Fixed Size:
                               ( ) Fill maximum size of (MB): 1____
                                ( ) Fill all available space:

                     [ ] Force to be a primary partition
                        [ ] Check for bad blocks

               OK                              Cancel

    F1-Help     F2-New      F3-Edit   F4-Delete    F5-Reset    F12-OK
```

9. The ‘Partitioning’ screen reappears. Press the **Tab** key until **OK** ishighlighted. Press **Enter**.



```
Red Hat Linux (C) 2003 Red Hat, Inc.

                            Partitioning

       Device          Start   End    Size       Type     Mount Point
  /dev/hda
    /dev/hda1              1    261   2047M  ntfs
    /dev/hda2            262   1000   5796M  ext3         /
    /dev/hda3           1001   1033    258M  swap
    Free space          1034   1044     86M  Free space

     New      Edit     Delete     RAID     OK     Back

    F1-Help     F2-New      F3-Edit   F4-Delete    F5-Reset    F12-OK
```

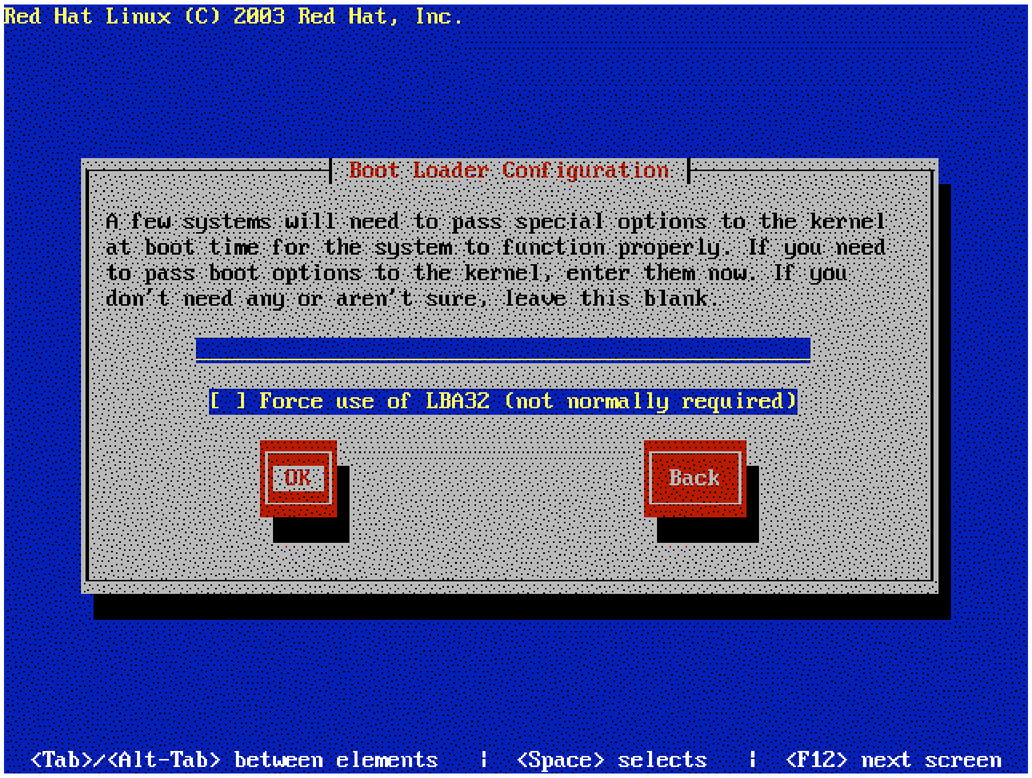
10. The ‘Boot Loader Configuration’ screen appears. Press the **Tab** key until **OK** is highlighted, and then press **Enter**.



```
Red Hat Linux (C) 2003 Red Hat, Inc.

                 Boot Loader Configuration
     Which boot loader would you like to use?
               (*) Use GRUB Boot Loader
               ( ) Use LILO Boot Loader
                  ( ) No Boot Loader
```

OK
Back
<Tab>/<Alt-Tab> between elements | <Space> selects | <F12> next screen

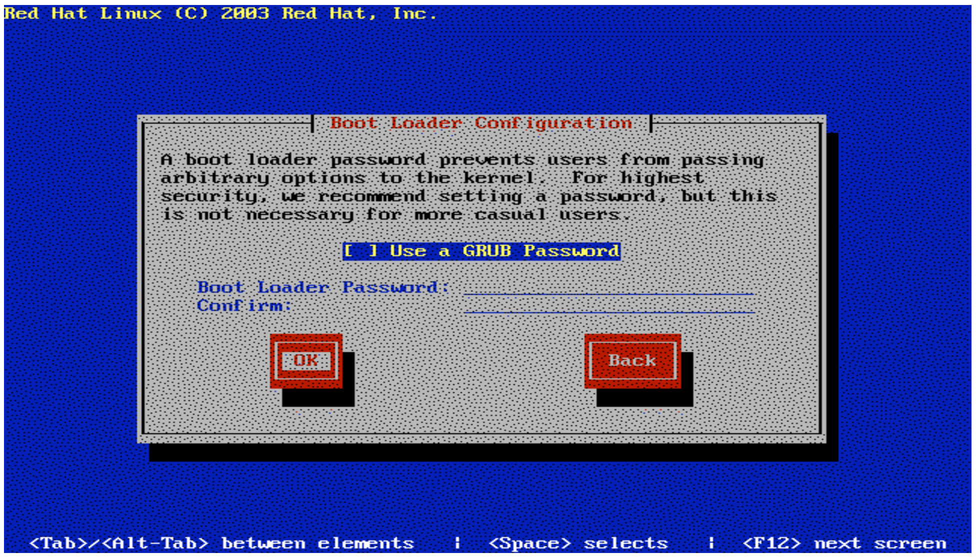
11. The ‘Boot Loader Configuration’ screen appears. Press the **Tab** key until **OK** is highlighted, and then press **Enter**.



```
Red Hat Linux (C) 2003 Red Hat, Inc.

                 Boot Loader Configuration
 A few systems will need to pass special options to the kernel
 at boot time for the system to function properly. If you need
 to pass boot options to the kernel, enter them now. If you
 don't need any or aren't sure, leave this blank.

      [ ] Force use of LBA32 (not normally required)
```

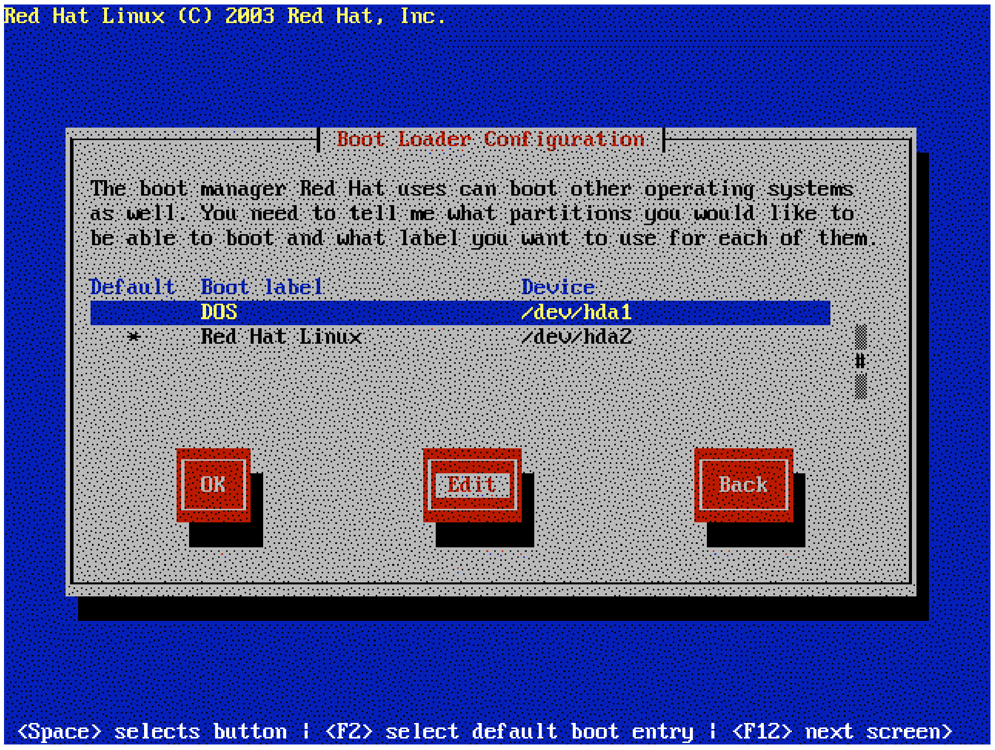
OK
Back
<Tab>/<Alt-Tab> between elements | <Space> selects | <F12> next screen

12. The ‘Boot Loader Configuration’ screen appears. Press the **Tab** key until **OK** is highlighted, and then press **Enter**.



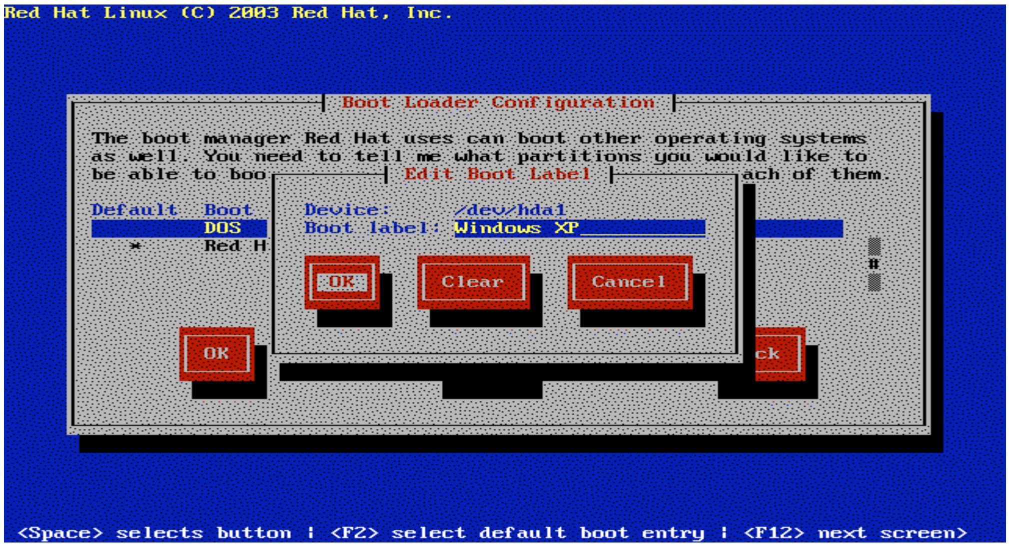
```
Red Hat Linux (C) 2003 Red Hat, Inc.

                 Boot Loader Configuration

   A boot loader password prevents users from passing
   arbitrary options to the kernel.  For highest
   security, we recommend setting a password, but this
   is not necessary for more casual users.

               [ ] Use a GRUB Password

      Boot Loader Password: ________________________
      Confirm:              ________________________

          OK                            Back

 <Tab>/<Alt-Tab> between elements  |  <Space> selects  |  <F12> next screen
```

13. The ‘Boot Loader Configuration’ screen appears. Use the **Tab** and arrowkeys to highlight **DOS** and then press the **Tab** key until **Edit** is highlighted, and then press **Enter**.



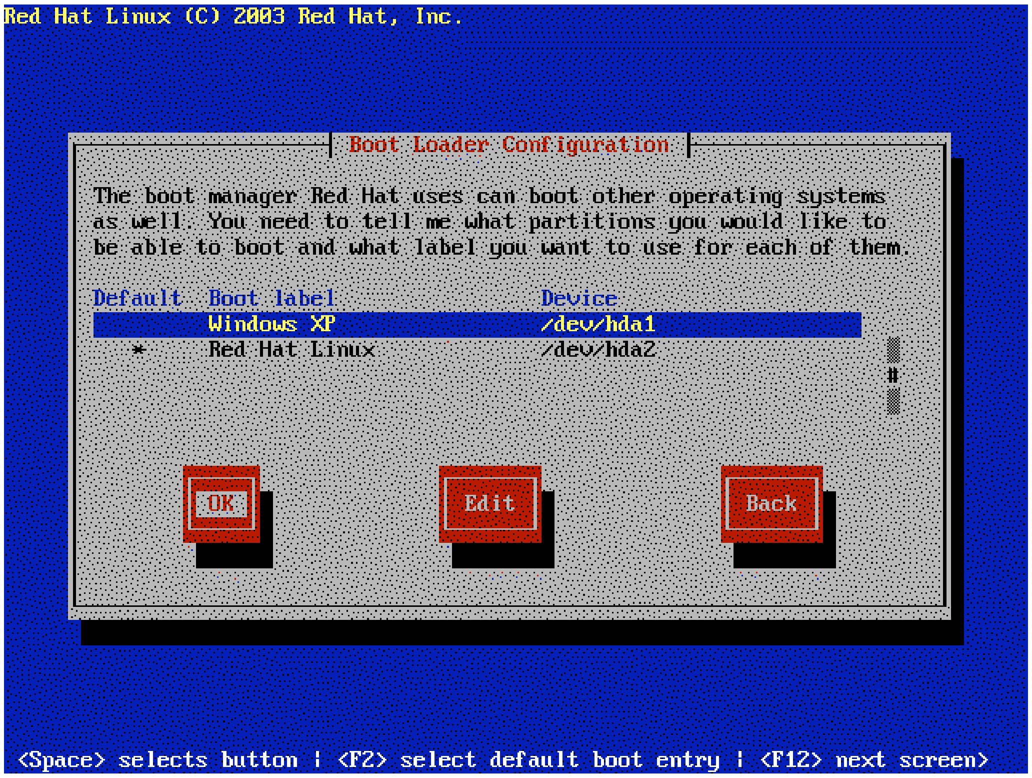
```
Red Hat Linux (C) 2003 Red Hat, Inc.

                 Boot Loader Configuration

   The boot manager Red Hat uses can boot other operating systems
   as well. You need to tell me what partitions you would like to
   be able to boot and what label you want to use for each of them.

   Default  Boot label                Device
            DOS                       /dev/hda1
      *     Red Hat Linux             /dev/hda2

         OK              Edit                Back

 <Space> selects button | <F2> select default boot entry | <F12> next screen
```

14. The 'Edit Boot Label' screen appears. Change the **Boot Label** field to**Windows XP.** Press the **Tab** key until **OK** is highlighted and press **Enter.**



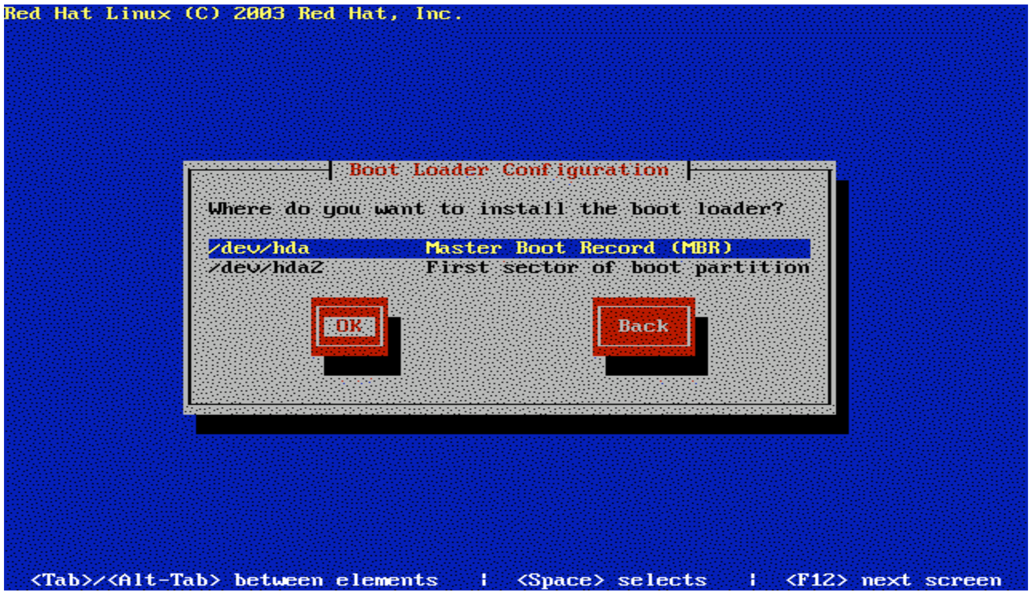
```
Red Hat Linux (C) 2003 Red Hat, Inc.

           ┤ Boot Loader Configuration ├
 The boot manager Red Hat uses can boot other operating systems
 as well. You need to tell me what partitions you would like to
 be able to boo┤ Edit Boot Label ├ach of them.

 Default  Boot   Device:       /dev/hda1
          DOS    Boot label:  Windows XP
    *     Red H
                  OK    Clear    Cancel

         OK                                    ck

 <Space> selects button | <F2> select default boot entry | <F12> next screen>
```

The 'Boot Loader Configuration' screen appears. Press the **Tab** key until **OK** is highlighted and press **Enter**.



```
Red Hat Linux (C) 2003 Red Hat, Inc.

           ┤ Boot Loader Configuration ├
 The boot manager Red Hat uses can boot other operating systems
 as well. You need to tell me what partitions you would like to
 be able to boot and what label you want to use for each of them.

 Default  Boot label                Device
          Windows XP                /dev/hda1
    *     Red Hat Linux             /dev/hda2

         OK             Edit             Back

 <Space> selects button | <F2> select default boot entry | <F12> next screen>
```

2. The ‘Boot Loader Configuration’ screen appears. Press the **Tab** key until **OK** is highlighted and press **Enter**.

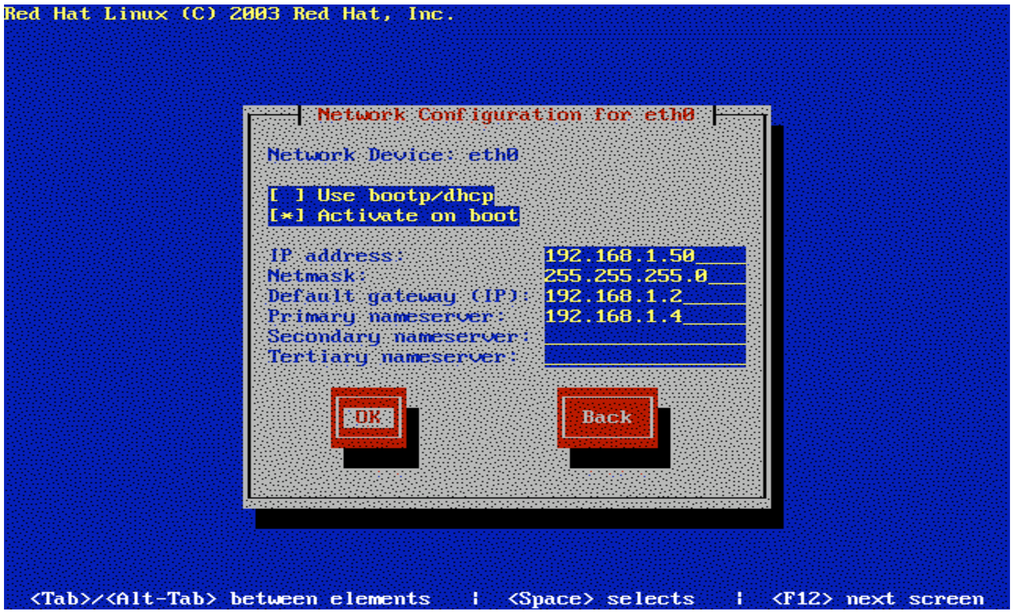


```
Red Hat Linux (C) 2003 Red Hat, Inc.

            Boot Loader Configuration
   Where do you want to install the boot loader?

   /dev/hda          Master Boot Record (MBR)
   /dev/hda2         First sector of boot partition

            OK                      Back

 <Tab>/<Alt-Tab> between elements  |  <Space> selects  |  <F12> next screen
```

3. The ‘Network Configuration for eth0’ screen appears. Press the **Spacebar**to remove the ***** (asterisk) in the following **[ ] Use bootp/dhcp** option. Press the **Tab** key to select the **IP address** field.

Enter the following parameters:

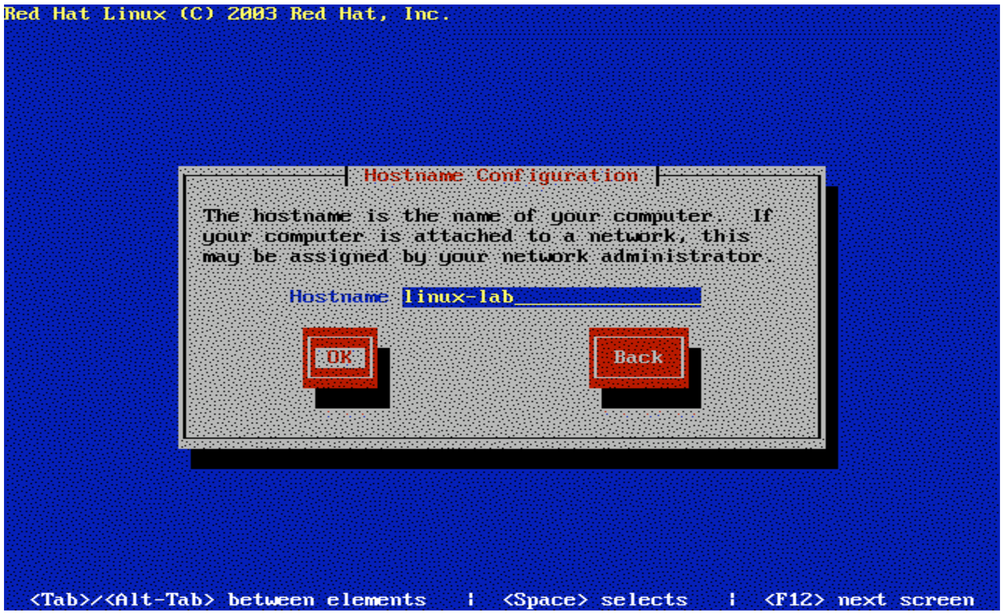
IP address:	192.168.1.50
Netmask:	255.255.255.0
Default gateway (IP):	192.168.1.2
Primary nameserver:	192.168.1.4



```
Red Hat Linux (C) 2003 Red Hat, Inc.

          Network Configuration for eth0
   Network Device: eth0

   [ ] Use bootp/dhcp
   [*] Activate on boot

   IP address:               192.168.1.50___
   Netmask:                  255.255.255.0__
   Default gateway (IP):     192.168.1.2____
   Primary nameserver:       192.168.1.4____
   Secondary nameserver:
   Tertiary nameserver:

         OK                    Back

 <Tab>/<Alt-Tab> between elements  |  <Space> selects  |  <F12> next screen
```

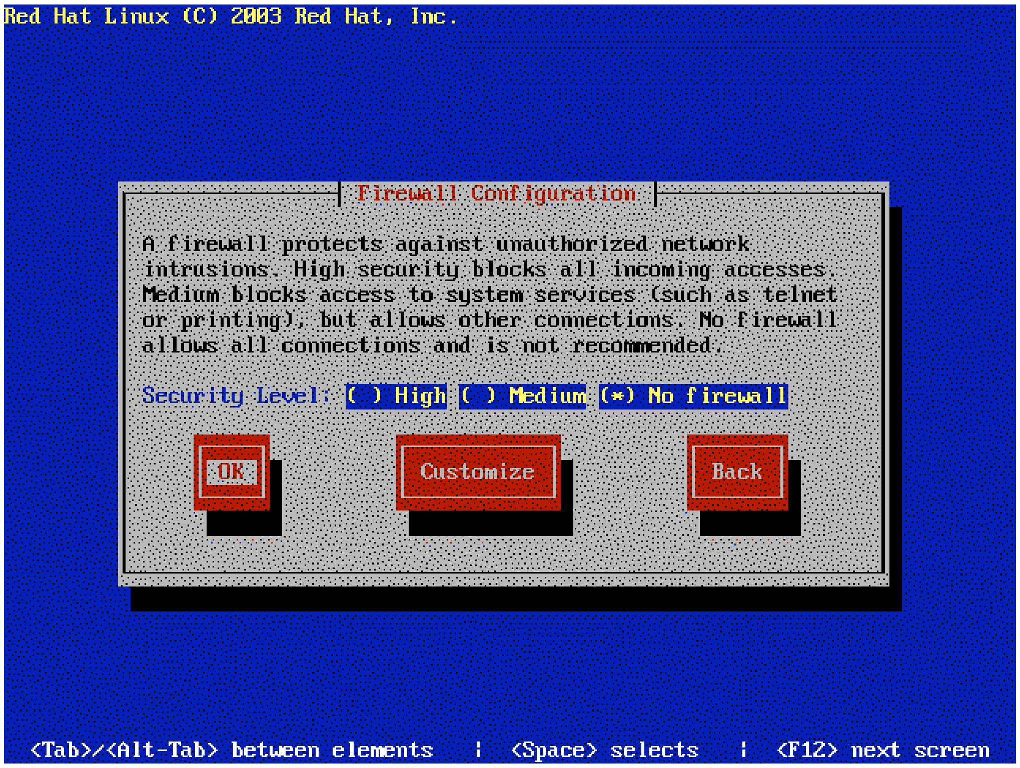
After you enter the parameters, press the **Tab** key until **OK** is highlighted, andthen press **Enter**.

4. The ‘Hostname Configuration’ screen appears. Enter **linux-lab** in the Hostname field. Press the **Tab** key until **OK** is highlighted, and then press**Enter**.



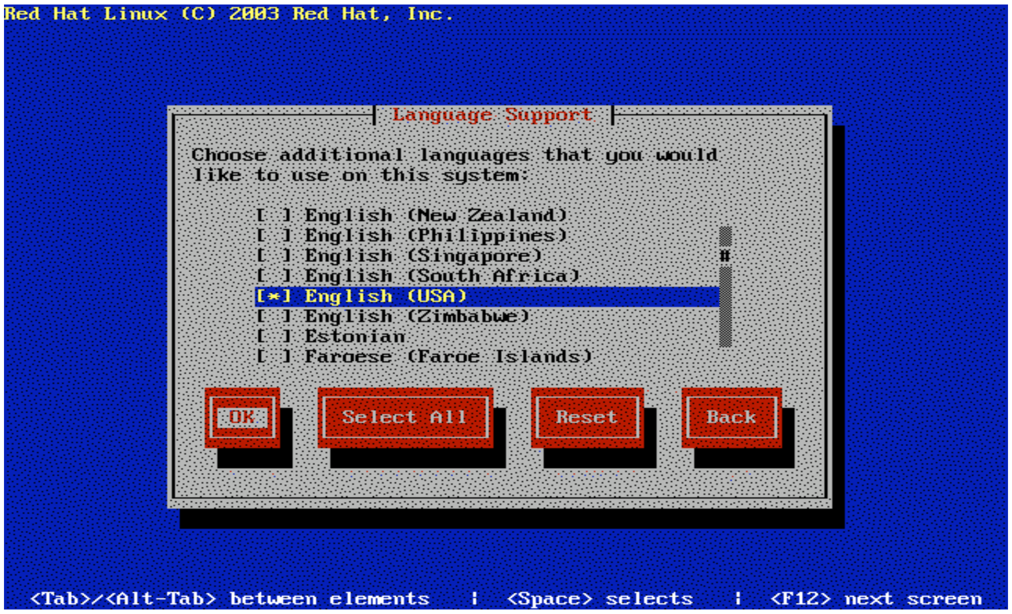
```
Red Hat Linux (C) 2003 Red Hat, Inc.

                Hostname Configuration

 The hostname is the name of your computer.  If
 your computer is attached to a network, this
 may be assigned by your network administrator.

        Hostname linux-lab_

          OK                          Back

 <Tab>/<Alt-Tab> between elements  |  <Space> selects  |  <F12> next screen
```

The ‘Firewall Configuration’ screen appears. Press the **Tab** key until **( ) No Firewall** is selected. Press the **Spacebar** to insert an asterisk (*), as shown in the screen. Then, press the **Tab** key until **OK** is highlighted andpress **Enter**.



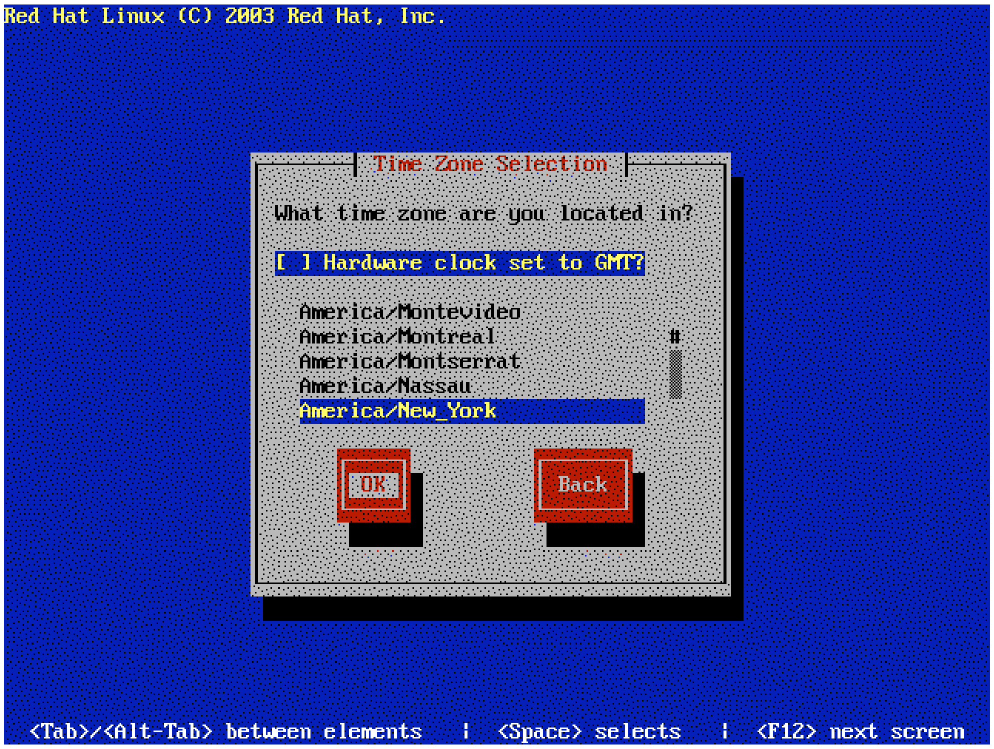
```
Red Hat Linux (C) 2003 Red Hat, Inc.

                Firewall Configuration

 A firewall protects against unauthorized network
 intrusions. High security blocks all incoming accesses.
 Medium blocks access to system services (such as telnet
 or printing), but allows other connections. No firewall
 allows all connections and is not recommended.

 Security Level: ( ) High ( ) Medium (*) No firewall

     OK            Customize            Back

 <Tab>/<Alt-Tab> between elements  |  <Space> selects  |  <F12> next screen
```

5. The ‘Language Support’ screen appears. Press the **Tab** key until **OK** is highlighted, and then press **Enter**.



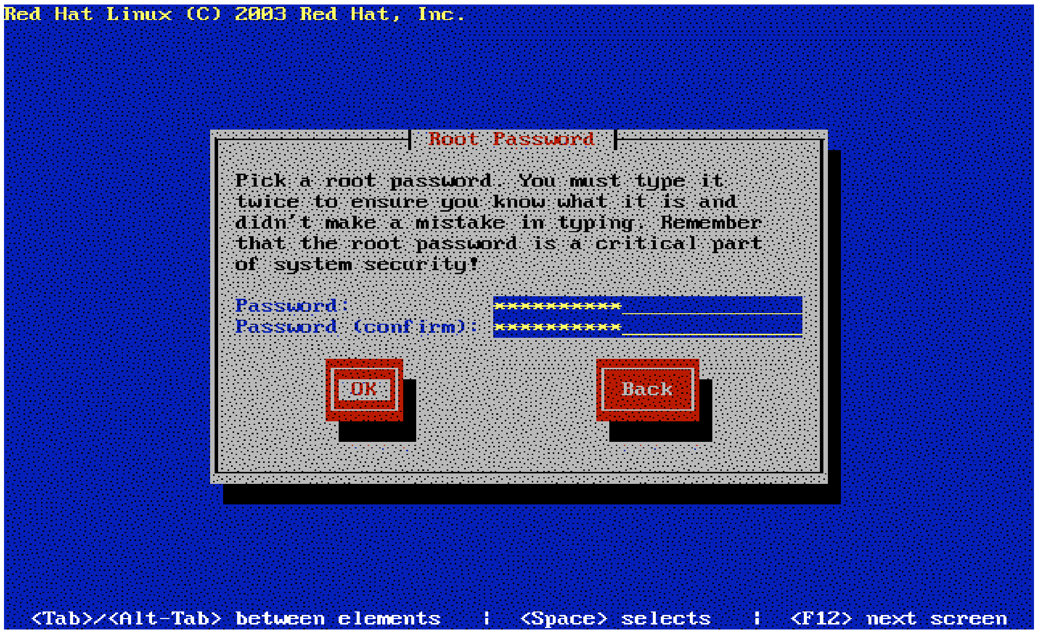
```
Red Hat Linux (C) 2003 Red Hat, Inc.

                    Language Support

    Choose additional languages that you would
    like to use on this system:

         [ ] English (New Zealand)
         [ ] English (Philippines)
         [ ] English (Singapore)
         [ ] English (South Africa)
         [*] English (USA)
         [ ] English (Zimbabwe)
         [ ] Estonian
         [ ] Faroese (Faroe Islands)

     OK     Select All     Reset     Back

 <Tab>/<Alt-Tab> between elements  |  <Space> selects  |  <F12> next screen
```

The ‘Time Zone Selection’ screen appears. Press the **Tab** key until **OK** is highlighted, and then press **Enter**. (If you are in a different time zone, usethe **Tab** and **arrow** keys to select the appropriate time zone.)



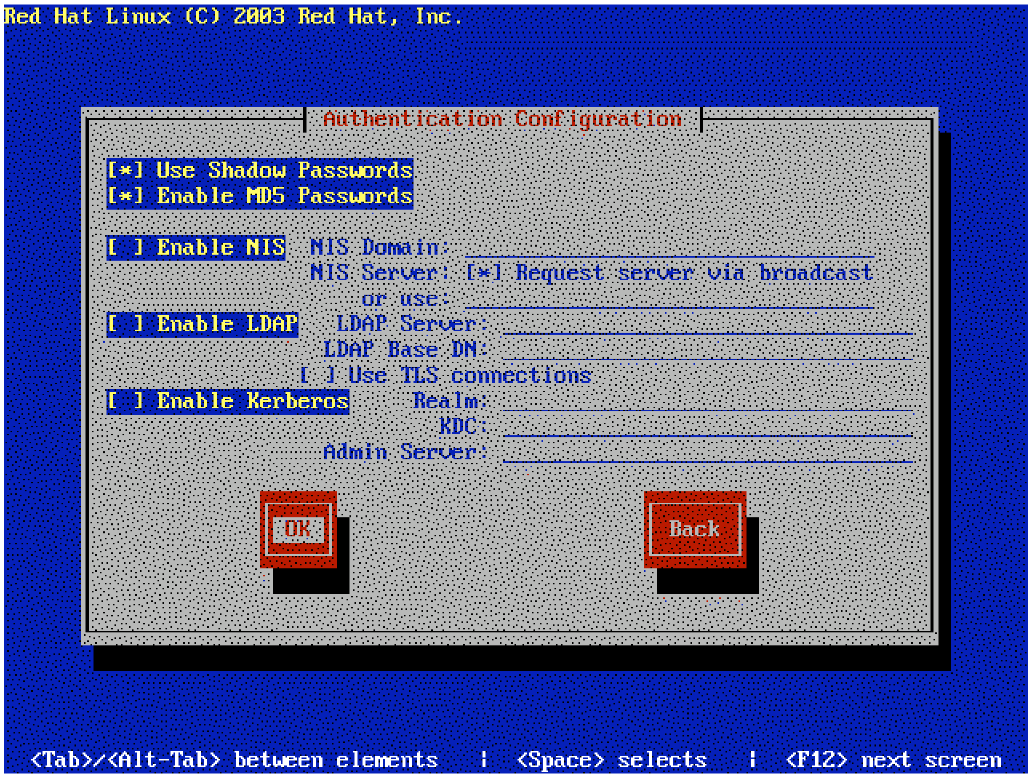
```
Red Hat Linux (C) 2003 Red Hat, Inc.

              Time Zone Selection

  What time zone are you located in?

  [ ] Hardware clock set to GMT?

    America/Montevideo
    America/Montreal
    America/Montserrat
    America/Nassau
    America/New_York

       OK               Back

 <Tab>/<Alt-Tab> between elements  |  <Space> selects  |  <F12> next screen
```

6. The 'Root Password' screen appears. In the **Password:** field, type a strong password to use for the root account. Confirm the password bytyping it in the **Password (confirm)**: field. Press **Tab** until **OK** is highlighted and press **Enter**.



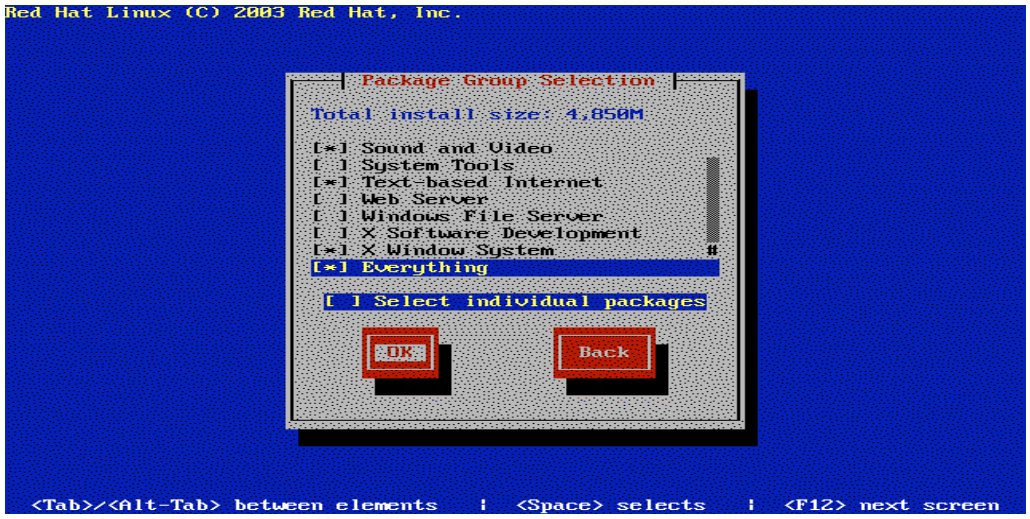
```
Red Hat Linux (C) 2003 Red Hat, Inc.

                 Root Password

  Pick a root password. You must type it
  twice to ensure you know what it is and
  didn't make a mistake in typing. Remember
  that the root password is a critical part
  of system security!

  Password:            **********
  Password (confirm):  **********

        OK                    Back

 <Tab>/<Alt-Tab> between elements  |  <Space> selects  |  <F12> next screen
```

The 'Authentication Configuration' screen appears. Press the **Tab** keyuntil **OK** is highlighted, and then press **Enter**.



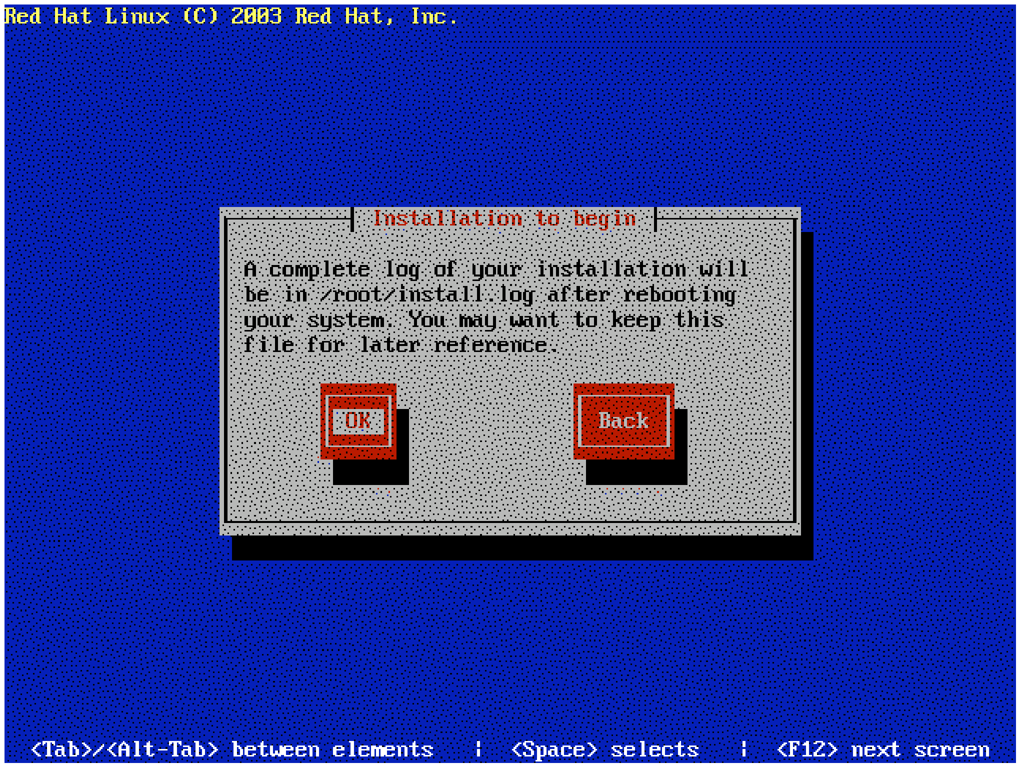
```
Red Hat Linux (C) 2003 Red Hat, Inc.

              Authentication Configuration

 [*] Use Shadow Passwords
 [*] Enable MD5 Passwords

 [ ] Enable NIS  NIS Domain: ________________________
                 NIS Server: [*] Request server via broadcast
                     or use: ________________________
 [ ] Enable LDAP  LDAP Server: ________________________
                 LDAP Base DN: ________________________
                [ ] Use TLS connections
 [ ] Enable Kerberos   Realm: ________________________
                         KDC: ________________________
                Admin Server: ________________________

        OK                          Back

 <Tab>/<Alt-Tab> between elements  |  <Space> selects  |  <F12> next screen
```

7. The 'Package Group Selection' screen appears. Press **End** to highlight **Everything** and then press **Space Bar** to select it. (an asterisk identifiesthe option as selected) Press the **Tab** key until **OK** is highlighted and press **Enter**.



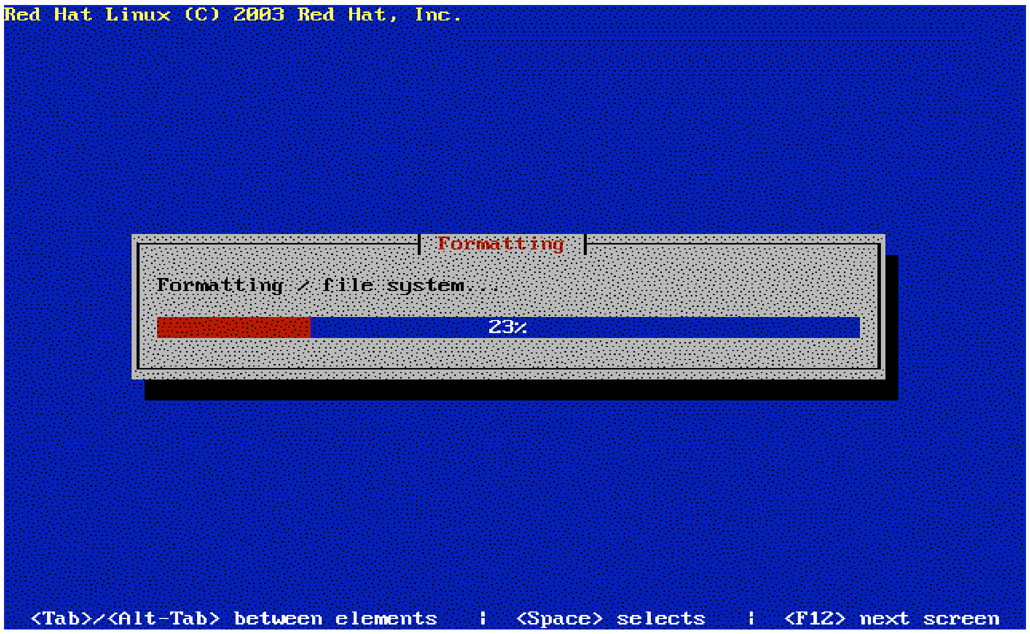
```
Red Hat Linux (C) 2003 Red Hat, Inc.

        Package Group Selection
  Total install size: 4,850M

  [*] Sound and Video
  [ ] System Tools
  [*] Text-based Internet
  [ ] Web Server
  [ ] Windows File Server
  [ ] X Software Development
  [*] X Window System
  [*] Everything

   [ ] Select individual packages

       OK            Back

 <Tab>/<Alt-Tab> between elements   |  <Space> selects   |  <F12> next screen
```

8. The 'Installation to begin' screen appears. Press **Enter**.



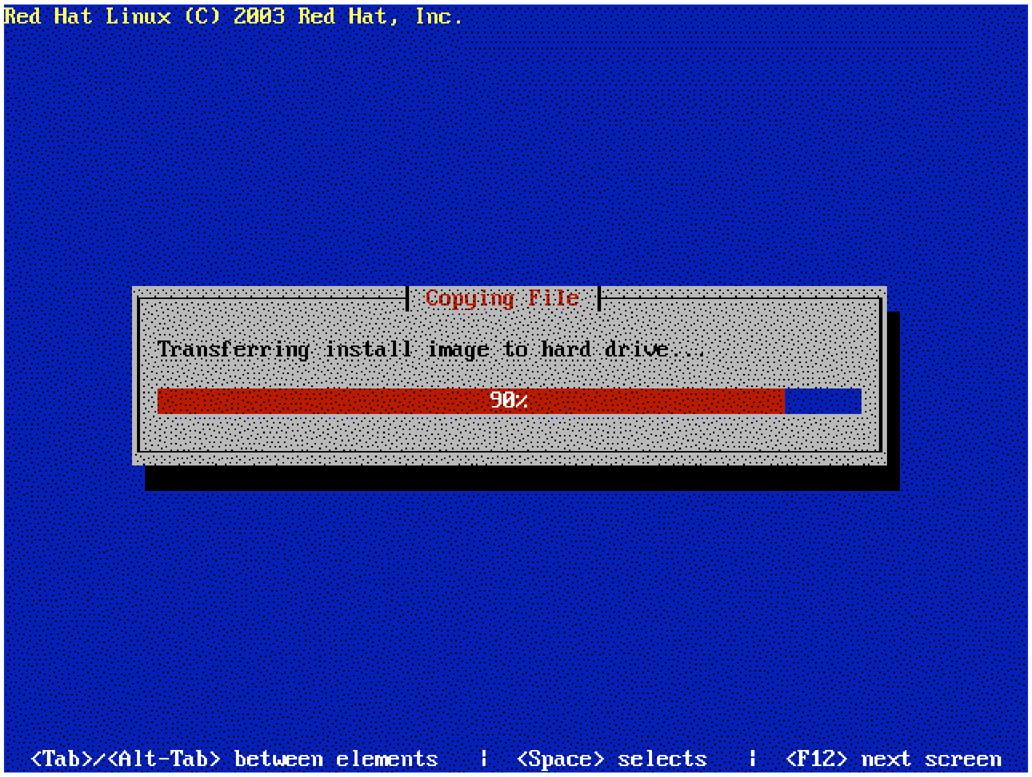
```
Red Hat Linux (C) 2003 Red Hat, Inc.

         Installation to begin

  A complete log of your installation will
  be in /root/install.log after rebooting
  your system. You may want to keep this
  file for later reference.

       OK                 Back

 <Tab>/<Alt-Tab> between elements   |  <Space> selects   |  <F12> next screen
```

9. The ‘Formatting’ screen appears. The **Formatting / file system…** message appears. Proceed to the next step.



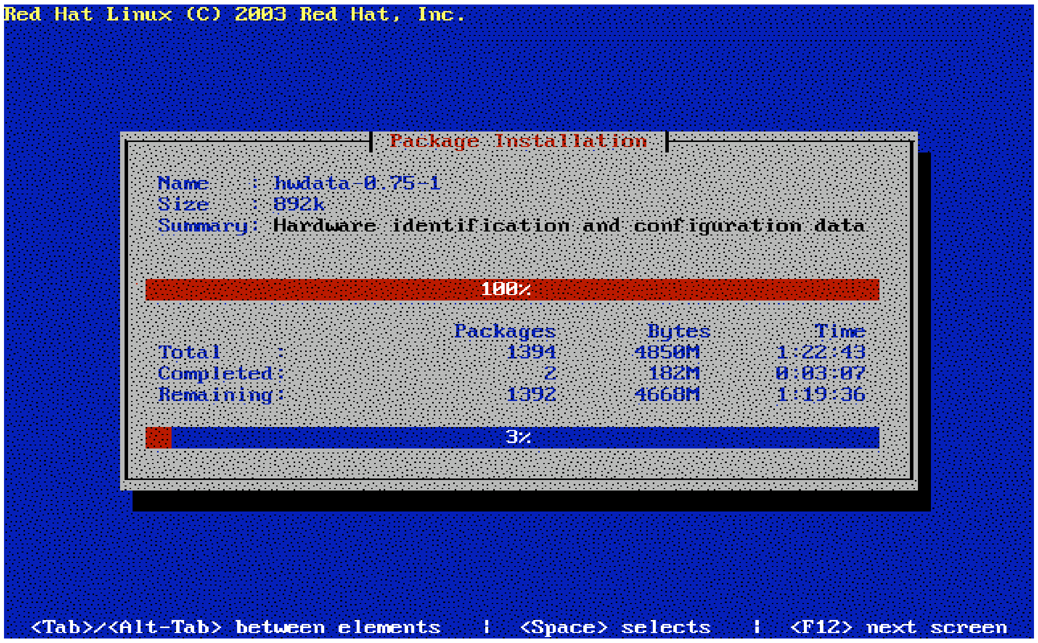
```
Red Hat Linux (C) 2003 Red Hat, Inc.

Formatting

Formatting / file system...
23%

<Tab>/<Alt-Tab> between elements   |  <Space> selects  |  <F12> next screen
```

The ‘Copying File’ screen appears. The **Transferring install image tohard drive…** message appears. Proceed to the next step.



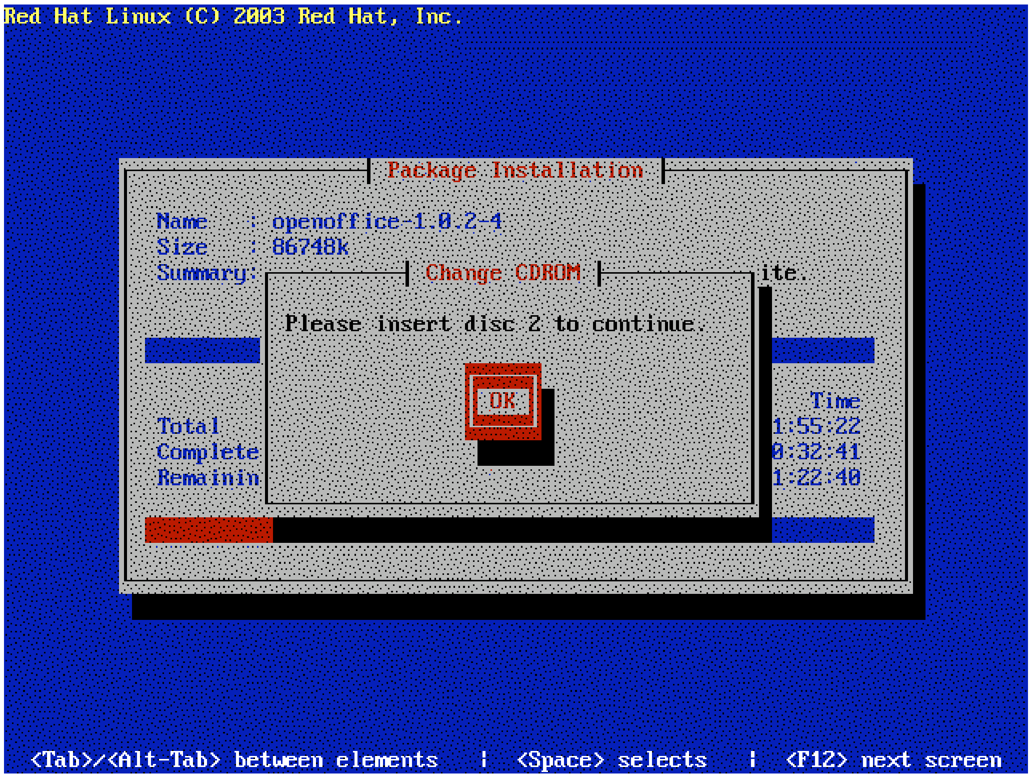
```
Red Hat Linux (C) 2003 Red Hat, Inc.

Copying File

Transferring install image to hard drive...
90%

<Tab>/<Alt-Tab> between elements   |  <Space> selects  |  <F12> next screen
```

10. The 'Package Installation' screen appears. Red Hat now starts installingthe packages. Proceed to the next step.



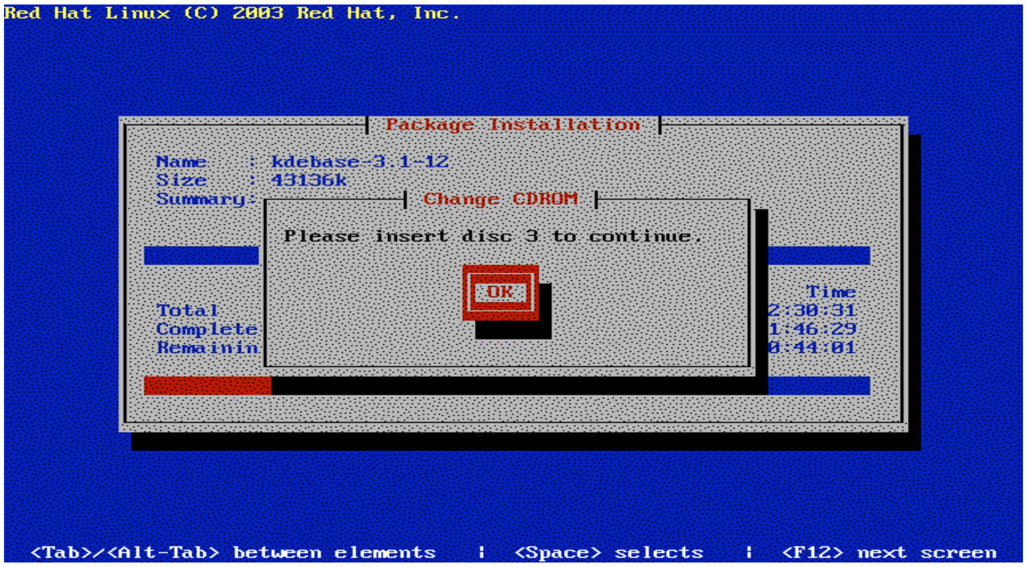
```
Red Hat Linux (C) 2003 Red Hat, Inc.

                ┤ Package Installation ├

  Name    : hwdata-0.75-1
  Size    : 892k
  Summary: Hardware identification and configuration data

                          100%

                           Packages       Bytes        Time
  Total     :                  1394       4850M     1:22:43
  Completed:                      2        182M     0:03:07
  Remaining:                   1392       4668M     1:19:36

                            3%

 <Tab>/<Alt-Tab> between elements  |  <Space> selects  |  <F12> next screen
```

The 'Change CDROM' screen appears. When prompted, insert the **RedHat Disc 2** and press **Enter**.



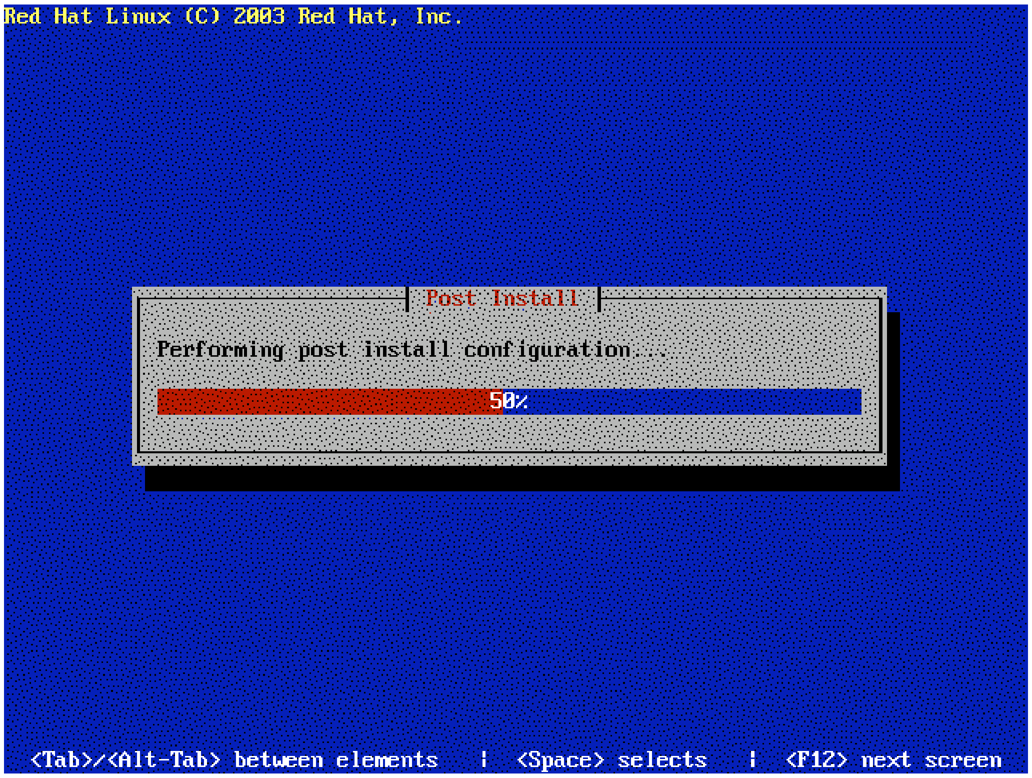
```
Red Hat Linux (C) 2003 Red Hat, Inc.

                ┤ Package Installation ├

  Name    : openoffice-1.0.2-4
  Size    : 86748k
  Summary:         ┤ Change CDROM ├                ite.

           Please insert disc 2 to continue.

                          OK
                                                    Time
  Total                                          1:55:22
  Complete                                       0:32:41
  Remainin                                       1:22:40

 <Tab>/<Alt-Tab> between elements  |  <Space> selects  |  <F12> next screen
```

11. The 'Change CDROM' screen appears again. When prompted, insert **RedHat Disc 3** and press **Enter**.



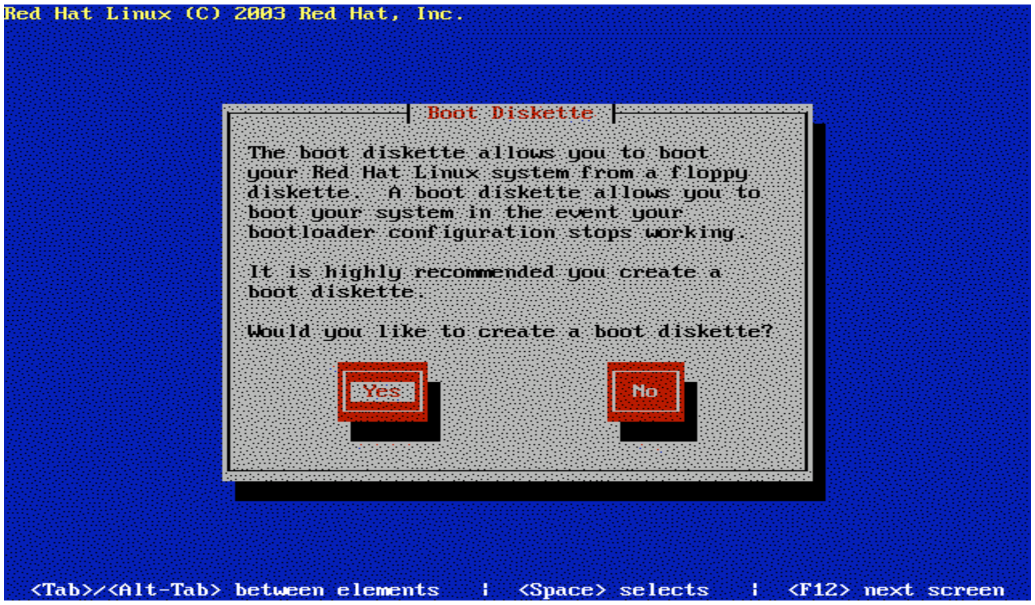
```
Red Hat Linux (C) 2003 Red Hat, Inc.

                    Package Installation
   Name    : kdebase-3.1-12
   Size    : 43136k
   Summary:        Change CDROM

            Please insert disc 3 to continue.

                          OK
                                                         Time
   Total                                              2:30:31
   Complete                                           1:46:29
   Remainin                                           0:44:01

 <Tab>/<Alt-Tab> between elements   |  <Space> selects  |  <F12> next screen
```

The 'Post Install' screen appears. After all of packages have been installed, Red Hat performs the post-install configuration, as shown in thefollowing screen. Proceed to the next step.



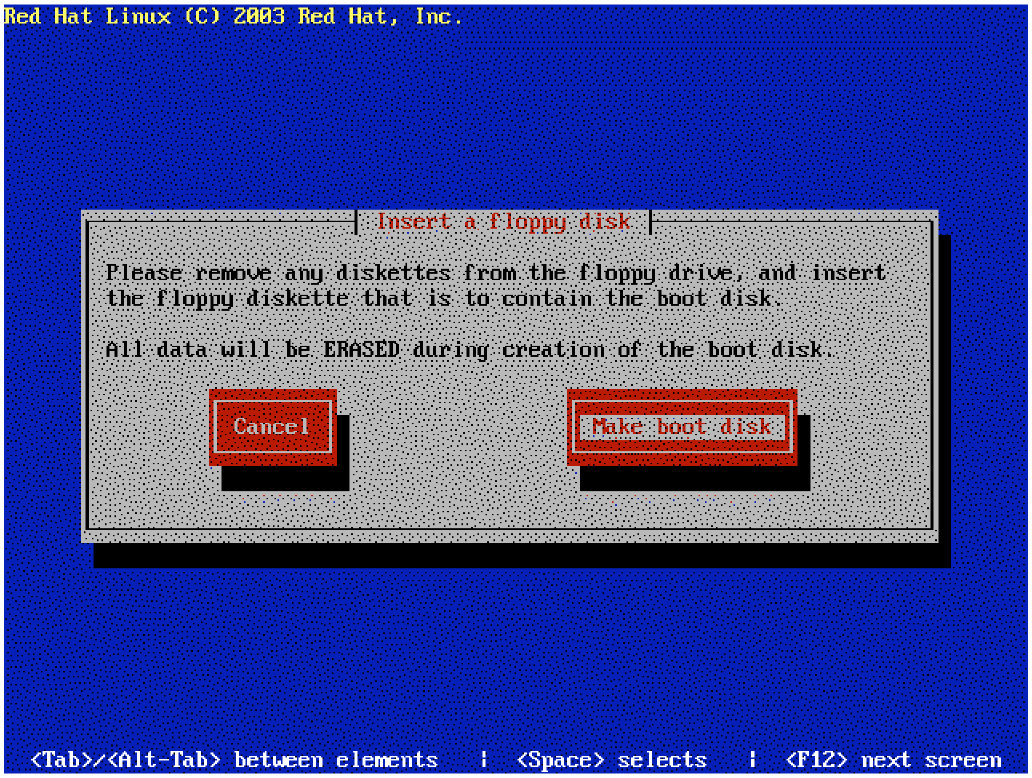
```
Red Hat Linux (C) 2003 Red Hat, Inc.

                    Post Install

   Performing post install configuration...

                         50%

 <Tab>/<Alt-Tab> between elements   |  <Space> selects  |  <F12> next screen
```

12. The ‘Boot Diskette’ screen appears. Press **Enter** to create a boot disk.



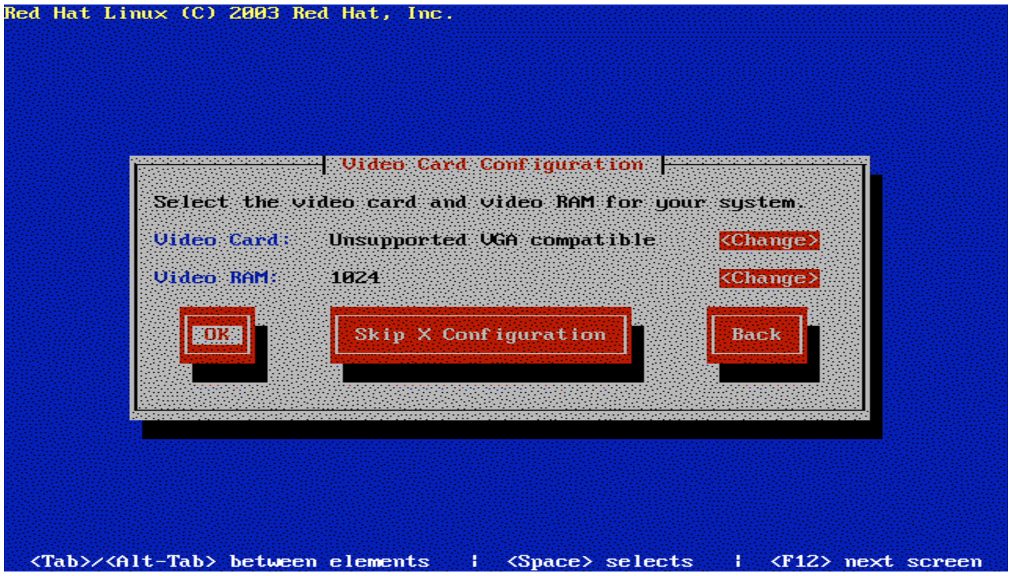
```
Red Hat Linux (C) 2003 Red Hat, Inc.

Boot Diskette

The boot diskette allows you to boot
your Red Hat Linux system from a floppy
diskette.  A boot diskette allows you to
boot your system in the event your
bootloader configuration stops working.

It is highly recommended you create a
boot diskette.

Would you like to create a boot diskette?

Yes          No

<Tab>/<Alt-Tab> between elements  |  <Space> selects  |  <F12> next screen
```

The ‘Insert a floppy disk’ screen appears. Insert a blank diskette into yourfloppy drive. Press **TAB** to highlight **Make boot disk** and press **Enter**.



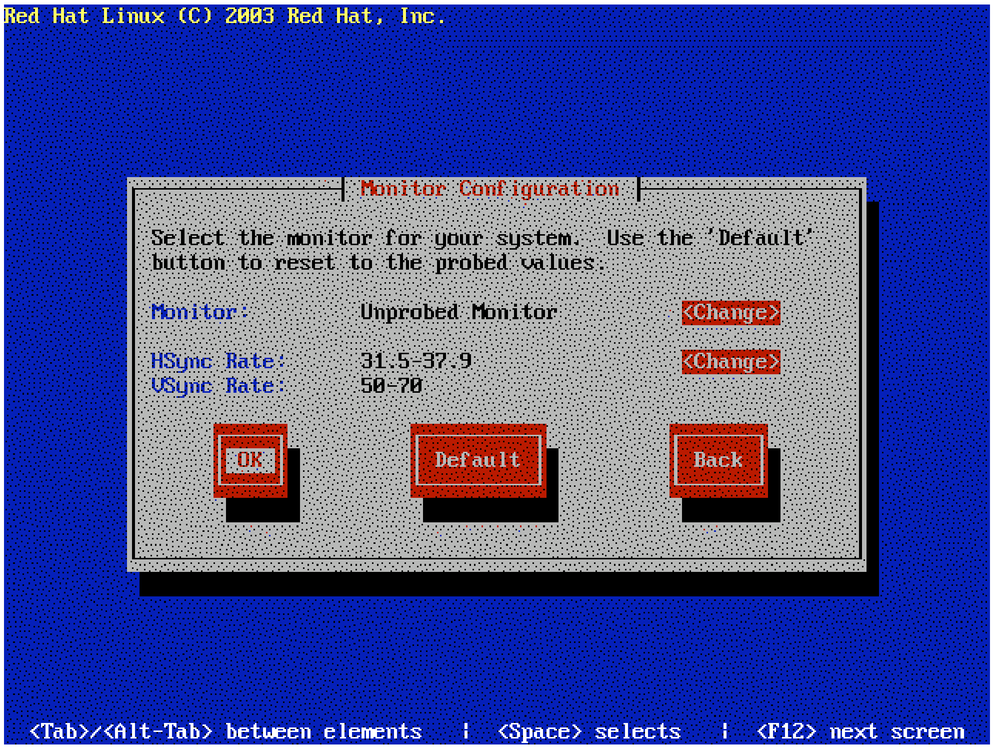
```
Red Hat Linux (C) 2003 Red Hat, Inc.

Insert a floppy disk

Please remove any diskettes from the floppy drive, and insert
the floppy diskette that is to contain the boot disk.

All data will be ERASED during creation of the boot disk.

Cancel          Make boot disk

<Tab>/<Alt-Tab> between elements  |  <Space> selects  |  <F12> next screen
```

13. The 'Video Card Configuration' screen appears. Use the **Tab** key and **Enter** key to select the appropriate video card settings for your system.Press the **Tab** key until **OK** is highlighted, and then press **Enter**.



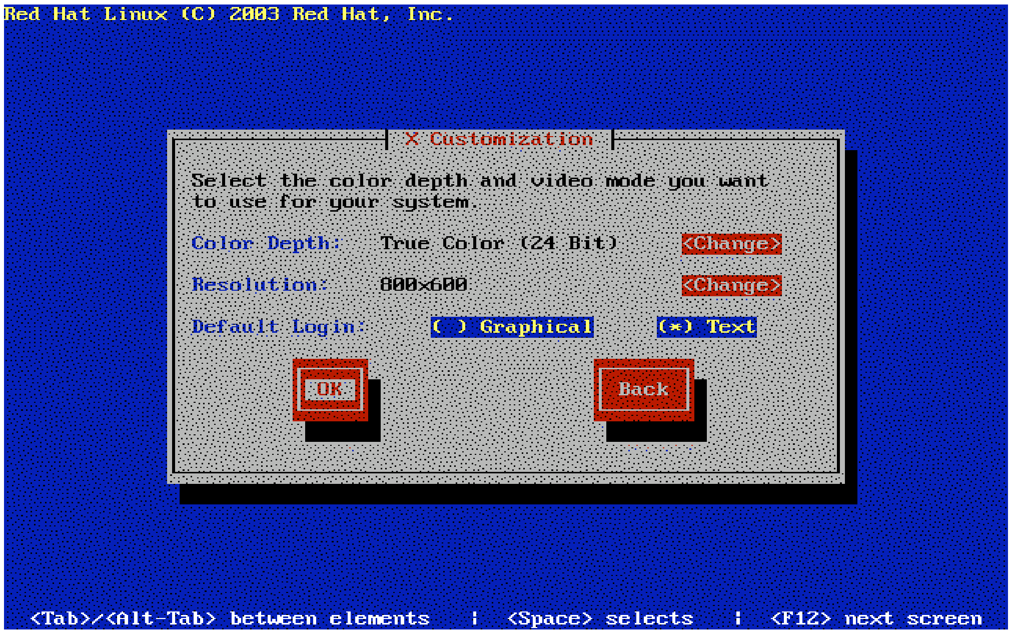
```
Red Hat Linux (C) 2003 Red Hat, Inc.

Video Card Configuration

Select the video card and video RAM for your system.
Video Card:    Unsupported VGA compatible    <Change>
Video RAM:     1024                          <Change>

<Tab>/<Alt-Tab> between elements  |  <Space> selects  |  <F12> next screen
```

The 'Monitor Configuration' screen appears. Again, use the **Tab** and **Enter** keys to select the appropriate monitor settings for your system.Press the **Tab** key until **OK** is highlighted, and then press **Enter**.



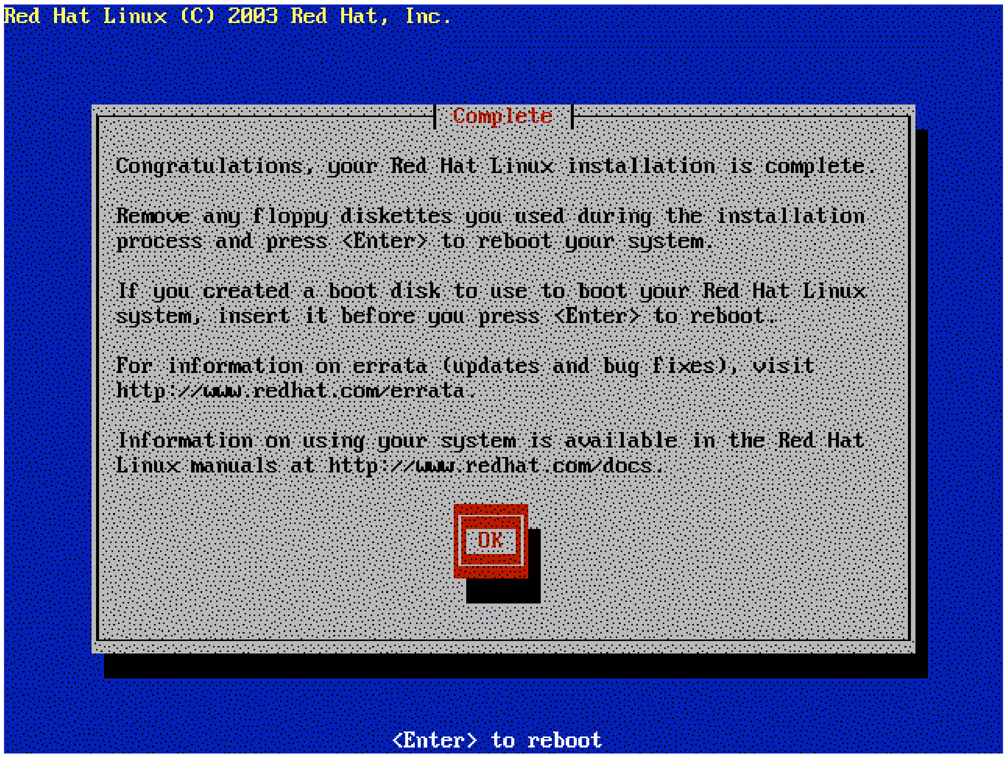
```
Red Hat Linux (C) 2003 Red Hat, Inc.

Monitor Configuration

Select the monitor for your system.  Use the 'Default'
button to reset to the probed values.

Monitor:       Unprobed Monitor     <Change>

HSync Rate:    31.5-37.9            <Change>
VSync Rate:    50-70

<Tab>/<Alt-Tab> between elements  |  <Space> selects  |  <F12> next screen
```

14. The ‘X Customization’ screen appears. Press the **Tab** key until **( ) Text** isselected, and then press the **Spacebar**. Press the **Tab** key until **OK** is highlighted, and then press **Enter**.



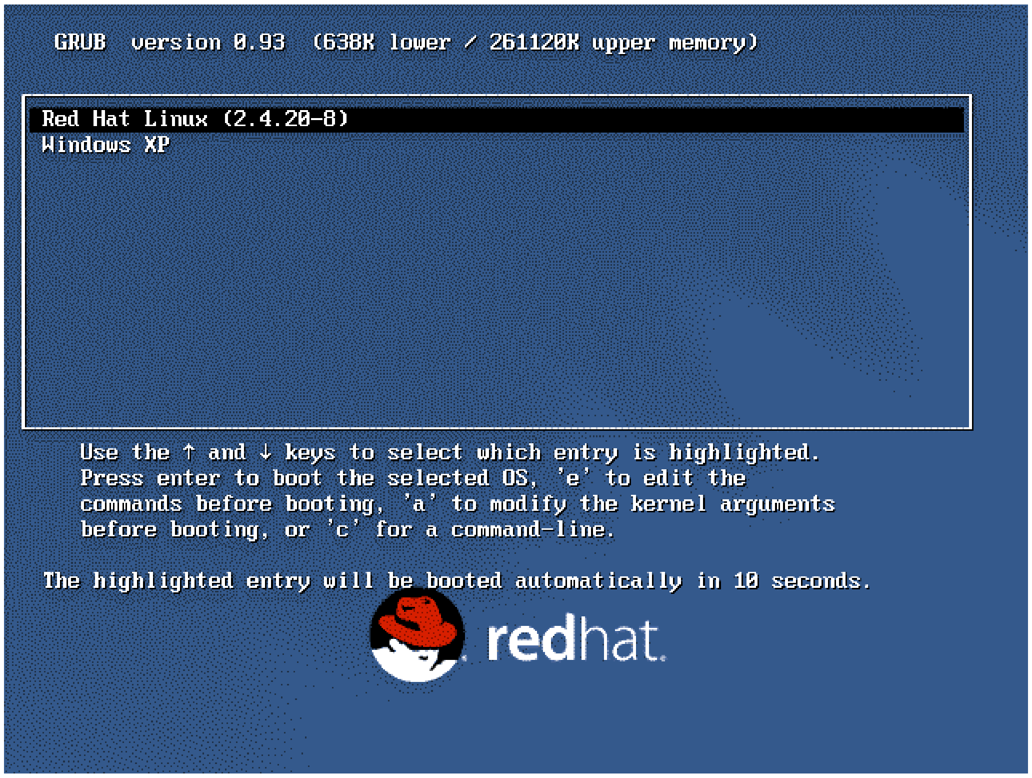
```
Red Hat Linux (C) 2003 Red Hat, Inc.

X Customization

Select the color depth and video mode you want
to use for your system.

Color Depth:     True Color (24 Bit)        <Change>

Resolution:      800x600                    <Change>

Default Login:      ( ) Graphical      (*) Text

       OK                      Back

 <Tab>/<Alt-Tab> between elements  |  <Space> selects  |  <F12> next screen
```

The ‘Complete’ screen appears. Congratulations, you have just installedRed Hat Linux. After removing the boot disk created earlier in the installation, press **Enter** to reboot the system. The CD-ROM will eject during the reboot process.



```
Red Hat Linux (C) 2003 Red Hat, Inc.

Complete

Congratulations, your Red Hat Linux installation is complete.

Remove any floppy diskettes you used during the installation
process and press <Enter> to reboot your system.

If you created a boot disk to use to boot your Red Hat Linux
system, insert it before you press <Enter> to reboot.

For information on errata (updates and bug fixes), visit
http://www.redhat.com/errata.

Information on using your system is available in the Red Hat
Linux manuals at http://www.redhat.com/docs.

                    OK

                <Enter> to reboot
```

15. As the system is rebooting you will be presented with the choice of booting into Red Hat or Windows XP. Use the arrow keys to select the OS that you want to boot into and press **Enter** to boot the choice. Note thatthe GRUB boot loader is only presented for a few seconds before the default OS is booted so you have to bepaying attention.



```
GRUB  version 0.93  (638K lower / 261120K upper memory)

Red Hat Linux (2.4.20-8)
Windows XP

   Use the ↑ and ↓ keys to select which entry is highlighted.
   Press enter to boot the selected OS, 'e' to edit the
   commands before booting, 'a' to modify the kernel arguments
   before booting, or 'c' for a command-line.

The highlighted entry will be booted automatically in 10 seconds.
```

RESULT:

Thus the installation of any guest operating system like Linux using VMware is done successfully.