Quaid-i-Azam University Islamabad

Final Project

PGD 10 (Semester III)

Submitted By:

Rana Zahida Nawaz

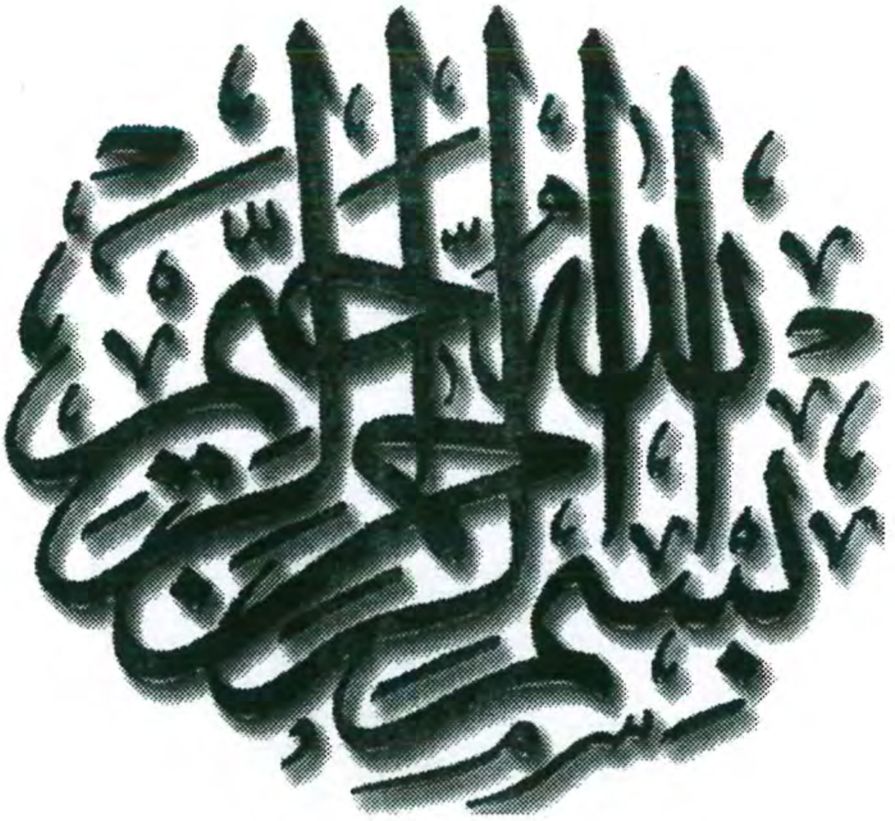
&

Fakhar-Zia-Kamal

Submitted To:

Dr. Ghulam Muhammad

Director of Computer Center (QAU)



*"IN THE NAME OF ALLAH,
MOST GRACIOUS , MOST MERCIFUL"*

COMPUTER AIDED EDUCATION OF PHYSICS PRACTICALS FOR HSSC_1 & 2

Supervised By

Dr. Ghulam Muhammad

Developed By

RANA ZAHIDA NAWAZ
&
FAKHAR ZIA KAMAL

Quaid-E-Azam University
Islamabad

COMPUTER AIDED EDUCATION OF PHYSICS EXPERIMENTS

By

Rana zahida nawaz

&

Fakhar zia kamal

A Project Report Submitted to
Quaid -i- Azam
University Islamabad.

In

Partial Fulfillment of the
Requirement of
Post Graduate Diploma in
Computer Sciences

Computer Center
Quaid –I- Azam University, Islamabad.

Computer Center Quaid –I- Azam
University, Islamabad.

Final Approval

This is to certify that we have read the project thesis submitted by Rana zahida nawaz and Fakhar zia kamal , and found it of sufficient standard to warrant its acceptance by the Quaid -i- Azam University, Islamabad for the Post Graduate Diploma in computer sciences.

COMMITTEE

EXTERNAL EXAMINER

NAME ______________________ **SIGNATURE:** ______________________

SUPERVISOR:

DR. GHULAM MUHAMMAD SIGNATURE: ______________________
COMPUTER CENTER
QAU ISLAMABAD

DIRECTOR:

DR. GHULAM MUHAMMAD SIGNATURE: ______________________
COMPUTER CENTER
QAU ISLAMABAD

DEDICATION

TO OUR PARENTS,
WHO ARE ALWAYS WITH US
TO LOVE, CARE AND ENCOURAGE.

PREFACE

This project is an effort to meet most of the requirements of FSc students . fundamental concepts have been explained in a simple and clear manner with the help of dynamic effects and computations of various experiments in physics.

Project contains a number of features to aid the students. Dynamic effects and calculations of observations are used to clarify the unfamiliar concepts. MCQ are included to help and guide the students through his or her study. it is also designed to stimulate and sustain the interest of students in physics. Viva voce questions are added for students

ACKNOWLEDGEMENTS

First and foremost, we thank to Allah the Beneficent and the Merciful, who always helps us to accomplish our goals in life.
In preparing this project we have had the benefit of comments by:

Our great appreciation and thanks to **Dr. GHULAM MUHAMMAD** for giving us an opportunity to take such a challenging task. He deserves our appreciation that contributed a lot making our project successful and meaningful.

Our heartiest gratitude and thanks are to **Mr. JAVED HUSSAIN** for his enduring support and guidance throughout the project.

We are thankful to our family for their support, which is always necessary for completion of our goals.

We are thankful to all our friends and class fellows for their working support and the creative ideas, which enhanced the beauty of our project and especially to our loving brother **RANA ZAHID ISLAM** for his cooperation and encouragement throughout our project.

Contents

page no

Chapter 1

Chapter 2

Chapter 3

Chapter 4

Chapter 5

Section 2:

Chapter 6

Chapter 7

Chapter 8

Chapter 1

INTRODUCTION

Man is created to learn & earn excellence. According to the heavenly message of AlQuraan "Iqra", we are advised to learn throughout our lives & explore the corners of the universe through knowledge. But learning any way requires teaching. In order to facilitate the process of learning teaching itself has gone through many developing stages & eventually emerged as a separate field of art. Technology is helping us in many ways---rather in all possible ways .

Purpose

In this era of technology revolution every face of our lives has been ornamented with the advancement of technology.

computers for example have brought revolution in the field of teaching also.we therefore chose our computer skills to be utilized in the field of teaching. Both of us being related to physics , we selected to prepare a computer based easy to learn presentation of our HSSC 1 & 2 practical courses.

Since computer technology has swept into all branches of arts & sciences, it should show miracles in the field of teaching also.since long there has been computer aided teaching programs in different fields but physics practical of intermediate & FSc levels in our

Usefulness

Since computer technology has swept into all branches of arts & sciences , it should show miracles in the field of teaching also.since long there has been computer aided teaching programes in different fields but physics practicals of intermediate & BSc levels in our country still required attention of computer experts.Though we don't call ourselves experts to that extent but we hope that this effort will prove to be very helpful to the students in this country in understanding their practicals & performing them in a better way.It will be a great privilege for us if our present effort (which we shall continue to update InshaAllah) prove to be a droplet in the ocean of computer technology.

Our enthusiasm and interest in the field of physics led our way, gradually towards success. Starting with our previous knowledge of graphics we moved on toward designing & creating animations related to our topic. This was of course a very difficult & technical job. The first thing we required was suitable software having the facilities of designing, programming, presentation, & mathematical support.

Problems

As we began with our work on this challenging but interesting project we had only a rough idea in mind as what to do but knew nothing about how to carry it out. We even didn't know about how to start with.

country still required attention of computer experts. Though we don't call ourselves experts to that extent but we hope that this effort will prove to be very helpful to the students in this country in understanding their practical & performing them in a better way. It will be a great privilege for us if our present effort (which we shall continue to update INSHAA ALLAH) proves to be a droplet in the ocean of computer technology.

BASIC AIMS OF OUR PROJECT

When we selected our project our .our basic aim Was to make a soft ware through which a student can easily learn how to perform experiments in laboratory.

REQUIRMENTS OF OUR PROJECT

Our requirements was as follows:

Title of project is computer aided education in physics experiments.

For this we have choose intermediate level physics experiments.

- 1 requirement of the project was to show dynamic effects in the experiments that a student can easily understand the experiments.
- Second requirement was that for computations we use some computer language which is able to run in window environment.
- Third was to record readings and observations
- The user may insert any value and using certain related formulas , may produce desired results.
- finally it calculate the formula for any experiment.
- Fourth was to make a quiz or MCQ for self test of the student it can help someone to see his knowledge about physics topics.
- Fifth was to make presentations and join all the work at one point.
- Sixth was to make viva voce questions which are mostly asked in the viva related to theory of the experiments.

Exploration for graphics software

We need graphics software to make the dynamic effects for this we explore lot of soft ware Such as xpower, animation shop, crocodile physics, circuit shop, and I use paintbrush fro making clips for dynamic effects

I did drawing in the ms word also for three dimensions so I try every one but the best and nice are animation shop and flash and circuit shop and lot of more. its needs lot of patience to make clips and the give them animations.

Its make the clips and adjust the size.

Then for loop and seen adjust every thing was really a tough job.

A small difference can create a lot of problem in dynamic effects.

So it needs lot of care and effort.

Chapter 2

Definition of physics

What is physics?

We can't give a one-sentence answer that does justice to this question.

we can say

Physics is the study of the laws of nature and their applications

Physics is the properties of matter, energy and of their mutual relationship. it tries to explain natural phenomena in terms of fundamental principles and laws.

Physics is the most basic of all sciences. it deals with the structure and matter behavior of matter. Physics is the body of knowledge gained from the study of natural phenomena.

Physics is what physicists do.

How physics is classified into different subjects?

The earlier divisions of physics were made on the basis of natural phenomena to which the methods of physics had been applied.

These branches consist of

- Classical mechanics
- Heat
- Sound
- Optics
- Electricity and magnetism etc.

Some times the physics is classified according to the particular aspect of study of nature examples are particle physics, atomic physics, nuclear physics, etc

sometimes classifications is made on the basis of particular instrument or technique used such as x-rays diffraction , neutron diffraction ,mass spectroscopy etc

AIMS OF PHYSICS

- To produce theories in terms of which the behavior of objects in the physical world can be described.
- To check these theories for inconsistencies by means of experiments.
- **To make predictions from the theories about matters not yet investigated as means of guidance for future research.**
- To discover new ways of using the physical knowledge we have not only to provide useful applications but also to have the ways for more fruitful investigations.

Historical development in physics

Physics is one of the oldest and most highly organized of all the sciences. it has great contribution to the cultural and economic of our world developments

Physics has been in action in all ages, therefore historically it has been divided into the following five periods.

- Ancient physics (3000AC to 600AC)
- Physics of Islamic Era (600 AD to 1500 AD)
- New awakening in physics (1500 AD to 1700 AD)
- Classical physics (1700 Ad to 1890 AD)
- Modern physics (1890 to present)
- Islamic Era

In this period Greek books were translated into Arabic and a lot of scientific activity took place in Muslims countries.

Teaching of Islam and science

The basic principles of science are observations and reasoning. Islam stresses both .the very first verse of

HOLY QURAN revealed to the HOLY PROPHET (saw) states.

"Read; in the name of your lord who is the creator. He has created man from a clot. Read: and your lord is the most generous who teaches by the pen teaches man that He didn't know."

The importance given by God Almighty to reading, writing, and scientific knowledge is clear from the verse. The Holy Quran in so many verses has again and again the believers to observe and think.

Two are quoted here

"Surely in the creation of heavens and the earth and in the interchange of night and day and in the ships which snail through the sea and in the rain which God sends down from heaven giving life to the earth after its death and scattering over it all kinds of cattle and in the clouds that are made to serve between the heavens and the earth are signs for those who understand"

"Will they not observe the camels how they are created? And the heaven how it is raised? And they are set up?"

The two verses quoted above are the orders from God to observe think and these now are recognized as the basic principles of modern science. Islam thus laid down the foundation of modern scientific thought 1400 years before.

Numerous other verses can be quoted in which the holy Quran invites the readers to observe and think.

("Don't you see? Don't you think? Don't you plan?"

The holy prophet (SWA) told his followers that learning of knowledge is the duty of every Muslim man or women.

It was the result of the Quranic injunctions and the holy prophet's (SWA) pronouncement that the spirit of enquiry
Spread rapidly through out the Muslim world. and the science with experimentation become well established in the Islamic era as we shall describe in the next article.

CHAPTER 3

About physics practicals of HSSC—1 & 2

The practical list for HSSC physics course has recently been revised & improved. The course designers have tried their best to accommodate about every field of study in physics. We started our work from the mechanics part if the syllabus & proceeded gradually ahead.

List of Computerized Experiments

Basic instruments fro measurements

- Vernier calipers
- Screw gauge

Mechanics

- To find the resultants of two vectors by graphical method
- To verify first condition of equilibrium.
- To verify second condition of equilibrium
- To study the changes in potential energy
- To determine the value of g by a simple pendulum
- To determine the value of g by free fall method

Waves

- To determine the velocity of sound at 0 C by resonance tube apparatus
- Using end correction method.
- To determine the velocity of sound at 0 C by resonance tube apparatus
- By two positions of resonance.
- To determine the frequency of tuning fork by a Sonometer.

OPTICS

- To determine the focal length of a convex lens using parallax method
- To determine the focal length of a concave lens using a concave mirror
- To determine the refractive index of water using concave mirror
- To determine the refractive index a glass prism
- To setup a compound microscope and find its magnifying power

Electricity & magnetism

To determine the resistance of a galvanometer using post-office box.

To determine the resistance of a wire using post-office box.

Physical quantities

vectors

_ Are those physical quantities which are specified by both direction as well as magnitude.

Scalars

_Are those which are specified by magnitude but have no direction.

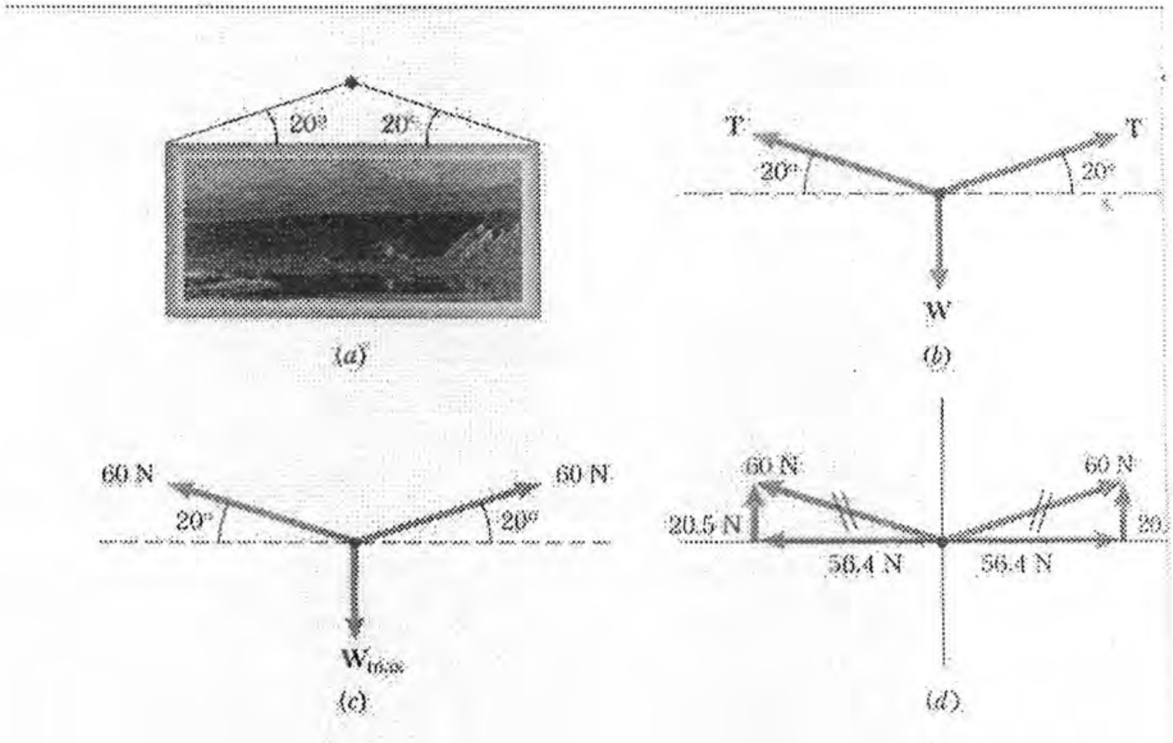
Physical quantities are divided into two main categories i.e; vectors & scalars.4.1 Vector addition

A hanging picture & its force diagram showing addition of vectors by rectangular components.

In part (a) of the picture the forces are shown by vectors T making an angle of 20° with the horizontal on both sides, whereas in (c) the forces are resolved into their components.

In order to find the resultant we have to find the sum of all the x-components as well as the sum of y-components. The resultant vector R is then obtained by the relation

$$R = R_X^2 + R_Y^2$$



Where

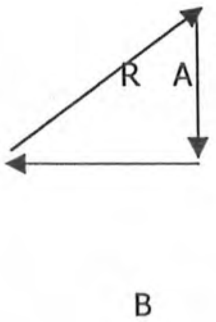
$$R_X = A_X + B_X$$

& $$R_Y = A_Y + B_Y$$

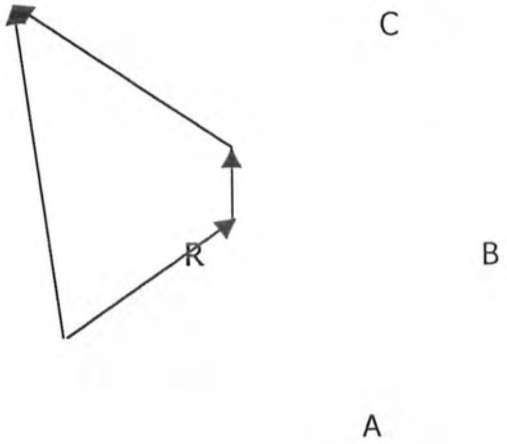
Where

A & B are any two vectors with rectangular components as A_X, A_Y & B_X , B_Y.

Addition of vectors by graphical method



(a) simple addition of two vectors



(b) addition of three vectors

We can find the resultant vectors of several vectors by using a scale diagram. This method for finding the resultant,called graphical metho, is easily to more than two vectors.-

The result of adding vector does not depend on the order in which we add them.

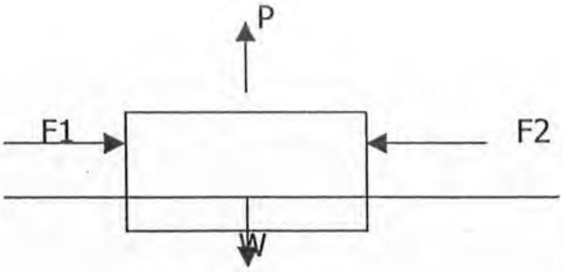
conditions of equilibrium

The branch of physics dealing with the systems at rest is called statics.It is of central importance to those who deal with designs & building of constructions. There are two conditions of equilibrium.

The first condition of equilibrium

For an object to be in equilibrium, the vector sum of the forces acting bon it must be zero. mathematically

$\Sigma F_x = 0$ $\Sigma F_y = 0$ $\Sigma F = 0$



A two-dimensional example is shown in the above figure. For the box to stay at rest with the four forces acting on it, the horizontal and vertical components of the forces must each add up to zero. Applying the above stated equations to this case gives us

$F_1 - F_2 = 0$ and $P - W = 0$

Here we have taken direction into account by supplying signs (right and up positive, left and down negative). The symbols F_1, F_2, W and P then represent the magnitudes of the forces.

The second condition of equilibrium

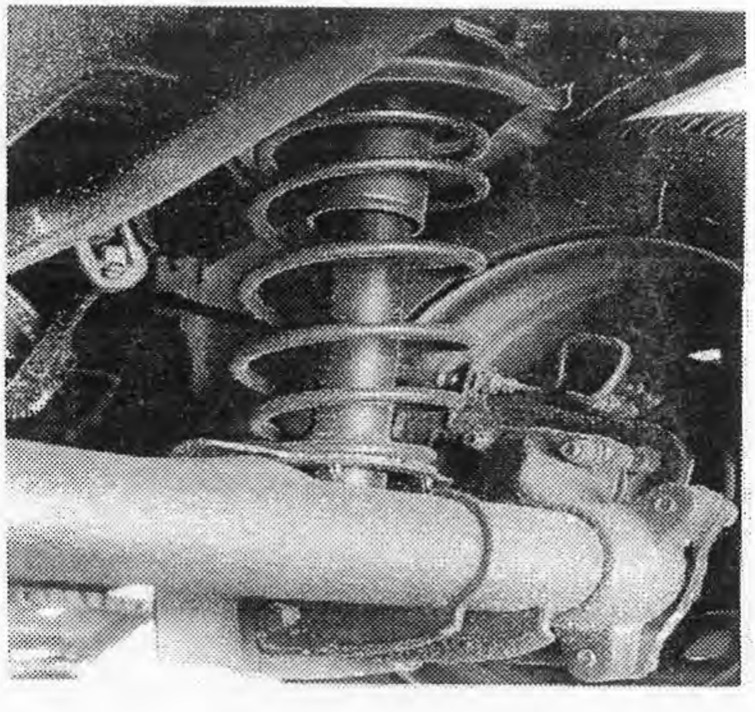
For an object to be in equilibrium, the algebraic sum of all the clockwise & anticlockwise torques acting on it must be zero.

Mathematically

$$\Sigma \tau = 0$$

Where τ is the symbol for torque.

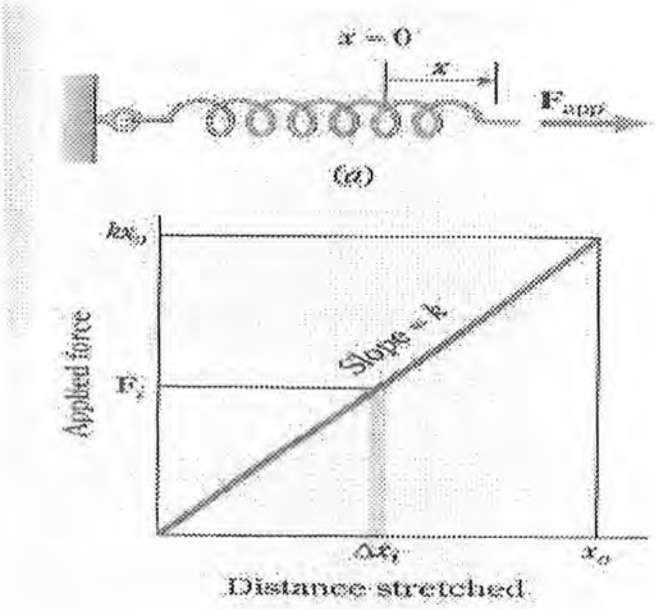
HOOKE'S LAW AND ELASTIC POTENTIAL ENERGY



We know that many elastic (spring like) systems obey Hook's law, which states that a distorting force is proportional to the distortion it causes. For a spring being stretched by an applied force F_{app}, as in the picture below, the displacement x the spring is stretched is related to F_{app} through

$F_{app} = \ \ kx$

where k is called the spring constant and has SI units of Newton's per meter.



The spring constant measures the "stiffness" of a spring. Larger values imply that larger forces are required to produce a given amount of stretch. The above figure shows how force varies with distortion for the spring. The graph is a straight line with slope k whose equation (Hooke's law) is given by the above equation.

Let us now compute the energy stored in a stretched or compressed spring _ that obeys Hooke's law.

We can show that the work done in stretching the spring from $x = 0$ to $x = x_o$ is the area under the graph line in Figure.. To do that, we examine the shaded rectangle shown. Its area is $F_i \Delta x_i$, where F_i is the stretching force that prevails during the small increase in distortion Δx_i . Because $W = F_S \Delta s$, this area is also the work done by the stretching force during this small increase in displacement. Imagine the region under the line from $x = 0$ to $x = x_o$ to be filled by many such rectangles. The sum of their areas gives the total work done in stretching the spring from $x = 0$ to $x = x_o$. Hence

The work done in stretching or compressing an elastic element is equal to the area under its F-versus-x graph line.

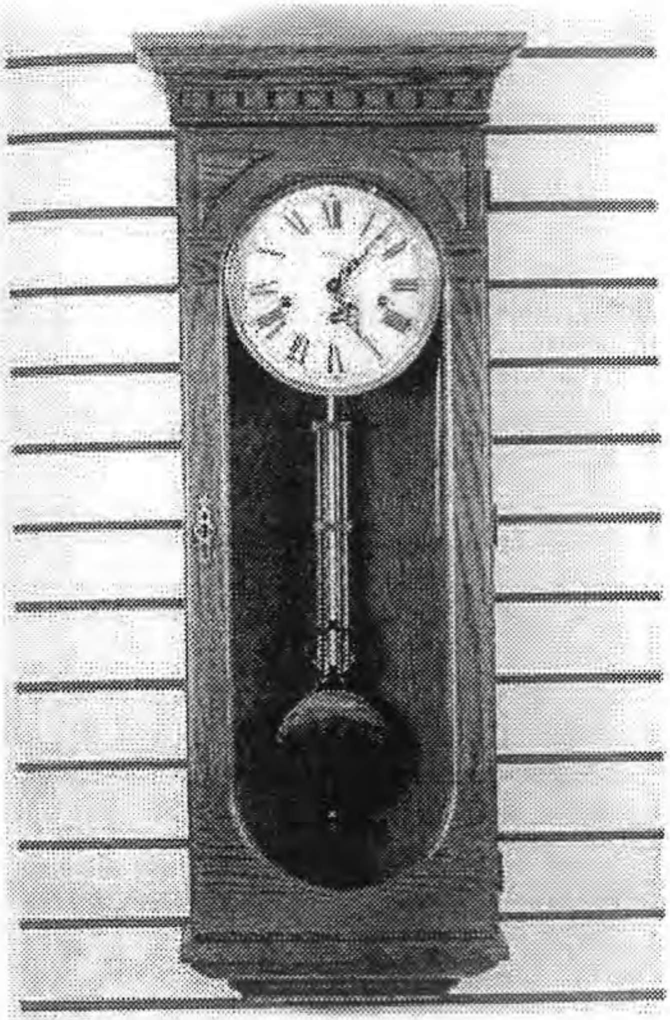
Since the area of a triangle is ½ base*height therefore the area under the graph line is

$$\tfrac{1}{2}\ Xo(K\, X_O)$$

This equals the work done in stretching the spring.

Thus elastic $PE = \tfrac{1}{2} K X^2$

Simple pendulum



A clock pendulum performs simple harmonic motion. Since the pendulum's period is constant, the clock is able to keep correct time.

A simple pendulum is a mass attached to a long string .the string is attached to a frictionless support so that it can oscillate freely .A simple pendulum performs simple harmonic motion. when such a pendulum is displaced from its mean position through a small angle θ it starts oscillating to & fro over the same path. The time period of a simple pendulum can be found by:

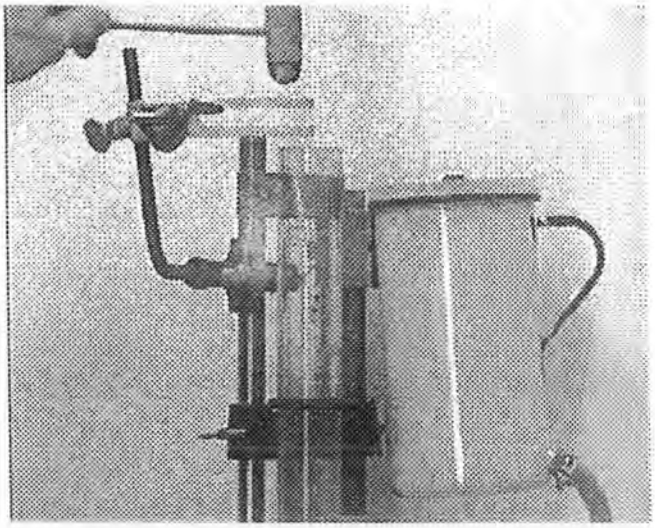
$T = 2\pi (l / g)^{1/2}$

Where T is the time period l is the length & g is the acceleration due to gravity.

Waves & sound

Resonance in air columns

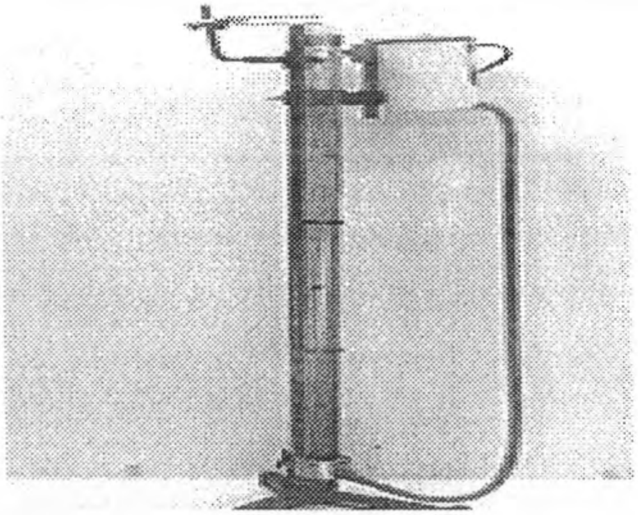
DESCRIPTION: A tuning fork mounted over the top of the tube is activated by striking it with a rubber hammer. Raising and lowering the reservoir varies the



SOUND RESONANCE IN WATER TUBE

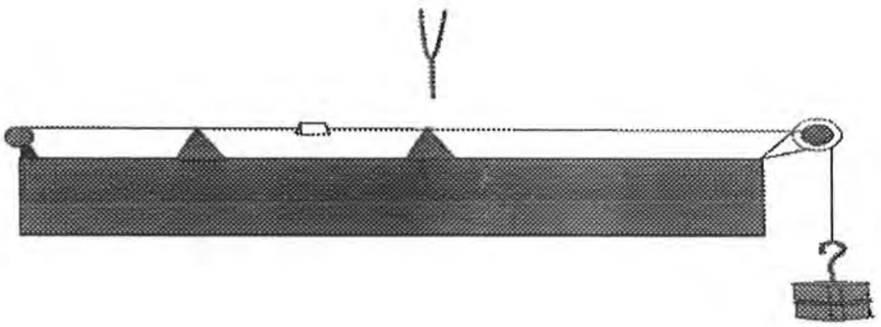
Water level in the tube to change the length of the air column. Because the air column is closed on one end (the surface of the water) resonance occur when the length of the tube is approximately 1/4, 3/4 or 5/4 of a wavelength, neglecting the end correction at the top of the tube. Using this apparatus standing waves can be demonstrated and the speed of sound determined to within about one percent.

EQUIPMENT: Air column - water reservoir device with tuning fork attached and mallet, as photographed.



The resonance tube apparatus

Sonometer



A sonometer is a wooden hollow box as shown above having a stretch string on the top across its ends .

The tension in the string can be varied by the suspended mass. When the stem of a vibrating tuning fork is touched to the board of the sonometer the string vibrates with the frequency of the tuning fork.

The formula for the frequency of the vibrating string is

$$f = (1 \ / \ 2 l) \ \ (T / m)^{1/2}$$

Optics

a. Spherical lenses

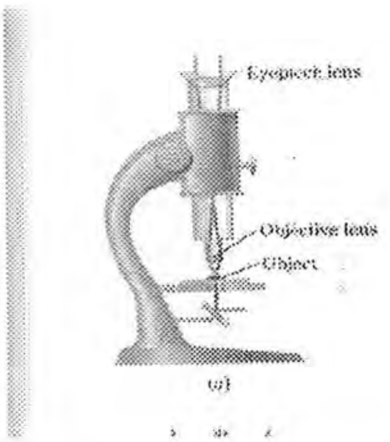
Lens is capable of focusing a beam of parallel light rays into a small region at a focal point. To see how Snell's law applies to the refraction of light incident upon a spherical surface.

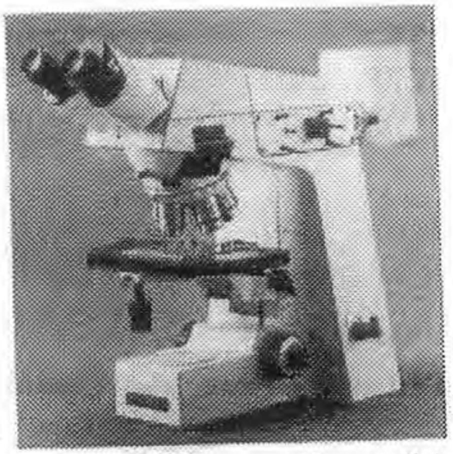
Lens formula

$$1 / f = 1 / p + 1 / q$$

Where f is the focal length , p is the distance between the lens and the object and q is the distance between the lens and the image.

b. Compound microscope





A compound microscope achieves grater magnification than a simple lens. A lens called the objective gives an a enlarged real image of an object placed very closed to it on the stage of the microscope. The eyepiece or ocular, makes further enlarged image of the real image formed by objective. The magnification of the compound microscope is given by the formula

$M = L d / fo\ fe$

Where d is the near point of the eye, fe is the focal length of eyepiece, fo is the focal length of objectives and L is the length of the microscope .

Electricity & magnetism

WHEATSTONE BRIDGE

A Whetstone bridge is connected as in the circuit below using two (approximately) 400 ohm resistors, a 0-90 ohm variable resistor and a 60 watt light bulb (16.7 ohms cold) with a 6 volt battery and a 0-5 mA ammeter (large lecture meter). The unknown resistance Rx (the light bulb) is:

$R_x = R_s (R_2/R_1)$,

Where Rs is the resistance of the active part of the variable resistor (equal to the fraction of its length used multiplied by 90 ohms).

Heat

Calorimeter

Many experiments involving heat are carried out in a container called calorimeter. This is a device that thermally isolates material so that heat cannot flow out of are into them from the surrounding. Heat is prevented from flowing through its double walls by the shiny metal coating on them and by the vacuum between them.

The sum of the heat exchanges within the calorimeter is zero.

In the words, the total energy of the isolated system inside the calorimeter is unchanged.

When a mass m undergoes a temperature change ΔT the amount of heat gained or heat lost is

$Q = m\,c\,\Delta T$

Where c is the specific heat of the material.

Chapter 4

TOOLS USED

ABOUT SOFTWARES WHICH I USED FOR DYNAMIC EFFECTS.

INFORMATION ABOUT THE ANIMATION SHOPE

Animation shop contains all the tools you need for creating, editing, and retouching your images. It is friendly enough for the casual user who wants to enhance family photographs, yet powerful enough for the professional who needs to create multi-layered graphics. This version, with the new vector layer features, gives Paint Shop Pro enhanced drawing capabilities.
Use the companion program, Jasc® Animation Shop™ 3.02, to create animations for a Web site, presentation, or multimedia publication. You can import animations into Paint Shop Pro for editing and return them to Animation Shop with the click of your mouse.

If this is the first time you have used Paint Shop Pro, browse through the help topics to get an idea of what the program can do. If you purchased the boxed version, you can also read the Getting Started and Reference Guides to familiarize yourself with the program.
You can learn more about using this help system and the other help resources in Using the Help Resources.

If you are familiar with previous versions of Paint Shop Pro, you can find a list of the new features in What's New in Paint Shop Pro 7. If you purchased the boxed version of the program, you can also read the Reference or Getting Started Guide for a more detailed description.

Use Animation Shop to create animations from one or more images and to enhance your animations with effects and transitions. You can now create multi-layered images in Paint Shop Pro and open them in Animation Shop with each layer becoming a separate frame. You can also use the Export command of Animation Shop to send animations to Paint Shop Pro. Each frame becomes a separate image that is linked back to the original animation. You can edit the images and then choose the Update Back to Animation Shop command. The changes are reflected in the original animation.

Opening Animation Shop

To open Animation Shop, choose File > Jasc Software Products > Launch Animation Shop or click the optional Launch Animation Shop button on the toolbar.

Use Animation Shop to create animations from one or more images and to enhance your animations with effects and transitions. You can now create multi-layered images in Paint Shop Pro and open them in Animation Shop with each layer becoming a separate frame. You can also use the Export command of Animation Shop to send animations to Paint Shop Pro. Each frame becomes a separate image that is linked back to the original animation. You can edit the images and then choose the Update Back to Animation Shop command. The changes are reflected in the original animation.

To import frames into Paint Shop Pro:

1 In Animation Shop, select individual frames or choose Edit > Select All to select the entire animation.

2 Choose File > Export Frames to Paint Shop Pro. Paint Shop Pro opens, and each frame is pasted into the program as an individual image with a transparent background and one layer.

As mentioned above, the new images are linked to their original frames. This makes it easy to apply any changes you make back to the original frames. Choose Edit > Update Back to Animation Shop, and the original animation is updated automatically.

You can also use the clipboard to paste one or more Animation Shop frames as Paint Shop Pro images. However, if you use this method, the images are not linked to their original frames. To apply changes to the original animation, you need to cut or copy each image and paste it back into its original frame.

To use the clipboard, select one or more frames of the animation and choose Edit > Copy. Open Paint Shop Pro, if necessary, and choose Edit> Paste AS Animation as Multiple Frames.

Use Animation Shop to create animations from one or more images and to enhance your animations with effects and transitions. You can now create multi-layered images in Paint Shop Pro and open them in Animation Shop with each layer becoming a separate frame. You can also use the Export command of Animation Shop to send animations to Paint Shop Pro. Each frame becomes a separate image that is linked back to the original animation. You can edit the images and then choose the Update Back to Animation Shop command. The changes are reflected in the original animation.

Using the Gradient Dialog Box

Choose this command or click the optional GIF Optimizer toolbar button to open the GIF Optimizer dialog box, where you can view the image as you configure the transparency, color reduction, and other format options. To create a transparency from a selection, make the selection before opening the dialog box. You can configure the settings in the tabbed pages or click the Use Wizard button and let Paint Shop Pro guide you through the process.

Using the GIF Optimizer Dialog Box

When you create a selection, you temporarily isolate part of an image so that you can edit it without affecting the rest of the image. You can move a selection, copy it, paint it, and apply special effects to it.

The selected area is surrounded by a border of black and white dashes called a marquee. The marquee defines the area of a selection, but is not restricted to it. You can move the marquee and the selection it encloses or move the marquee by itself. When you move only the marquee, the new area it surrounds becomes the new selection. The area no longer surrounded by the marquee is deselected. Selecting a vector object and floating it transforms it into a raster selection. You cannot defloat this selection back to its original vector layer because vector layers cannot contain raster information. When you defloat the selection, Paint Shop Pro places it on the nearest raster layer.

Selection Tools

You can create a selection by enclosing an area of the image, by selecting specific colors, or by pasting the contents of the clipboard as a selection. Paint Shop Pro provides three tools for creating selections:

- The Selection tool, which makes a selection of a specific shape,
- The Freehand tool, which makes a selection of an irregular shape, and
- The Magic Wand tool, which makes a selection of a range of color, hue, or brightness.

Creating a Selection

To create a selection:

1 Activate a selection tool by clicking its button on the Tool palette.

2 Select its options on the Tool Options palette.

3 Create the selection in the image.

Work in flash

Flash basics overview

Flash movies are graphics and animation for Web sites. They consist primarily of vector graphics, but they can also contain imported bitmap graphics and sounds. Flash movies can incorporate interactivity to permit input from viewers, and you can create nonlinear movies that can interact with other Web applications. Web designers use Flash to create navigation controls, animated logos, long-form animations with synchronized sound, and even complete, sensory-rich Web sites.

Flash movies are compact, vector graphics, so they download rapidly and scale to the viewer's screen size.

Animation in Flash

Using Flash, you can animate objects to make them appear to move across the Stage and/or change their shape, size, color, opacity, rotation, and other properties. You can create frame-by-frame animation, in which you create a separate image for each frame. You can also create tweened animation, in which you create the first and last frames of an animation and direct Flash to create the frames in between..

Creating animation overview

You create animation by changing the content of successive frames. You can make an object move across the Stage, increase or decrease its size, rotate, change color, fade in or out, or change shape. Changes can occur independently of, or in concert with, other changes. For example, you can make an object rotate and fade in as it moves across the Stage.

There are two methods for creating an animation sequence in Flash: frame-by-frame animation and tweened animation. In frame-by-frame animation you create the image in every frame. In tweened animation, you create starting and ending frames and let Flash create the frames in between. Flash varies the object's size, rotation, color, or other attributes evenly between the starting and ending frames to create the appearance of movement.

Tweened animation is an effective way to create movement and changes over time while minimizing file size. In tweened animation, Flash stores only the values for the changes between frames. In frame-by-frame animation, Flash stores the values for each complete frame.

Creating keyframes

A keyframe is a frame where you define changes in the animation. When you create frame-by-frame animation, every frame is a keyframe. In keyframe (tweened) animation, you define keyframes at important points in the animation and let Flash create the content of frames in between. Flash displays the interpolated frames of a tweened animation as light blue or green with an arrow drawn between key frames. Flash redraws shapes in each key frame. You should create key frames only at those points in the artwork where something changes.

Keyframes are indicated in the Timeline: a solid circle represents a key frame with content on it, and an empty key frame is represented by a vertical line before the frame. Subsequent frames that you add to the same layer will have the same content as the key frame.

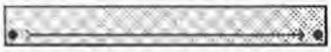
To create a keyframe, do one of the following:

- Select a frame in the Timeline and choose Insert > Key frame.
- Right-click (Windows) or Control-click (Macintosh) a frame in the Timeline and choose Insert Key frame.

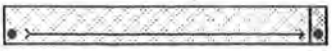
Representations of animations in the Timeline

Flash distinguishes tweened animation from frame-by-frame animation in the Timeline as follows:

- Motion-tweened key frames are indicated by a black dot and intermediate tweened frames have a black arrow with a light blue background.

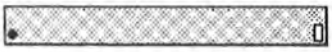


- Shape-tweened key frames are indicated by a black dot and intermediate frames have a black arrow with a light green background.

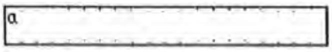


- A dashed line indicates that the final key frame is missing.

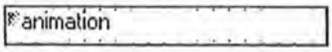
- A single key frame is indicated by a black dot. Light-gray frames after a single key frame contain the same content with no changes and have a black line with a hollow rectangle at the last frame of the span.



- A small a indicates that the frame has been assigned a frame action with the Actions panel.

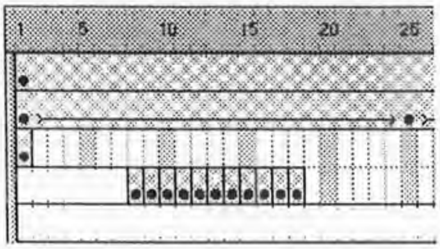


- A red flag indicates that the frame contains a label or comment.



About layers in animation

Each scene in a Flash movie can consist of any number of layers. As you animate, you use layers to organize the components of an animation sequence and to separate animated objects so they don't erase, connect, or segment each other. If you want Flash to tween the movement of several groups or symbols at once, each must be on a separate layer. Typically, the background layer contains static artwork. Additional layers contain one separate animated object each.



Layers appear as rows in the Timeline.

When a movie has several layers, tracking and editing the objects on one or two of them can be difficult. This task is easier if you work with the content one layer at a time.

About tweened animation

Flash can create two types of tweened animation. In motion tweening, you define properties such as position, size, and rotation for an instance, group, or text block at one point in time, and then you change those properties at another point in time. In shape tweening, you draw a shape at one point in time, and then you change that shape or draw another shape at another point in time. Flash interpolates the values or shapes for the frames in between, creating the animation.

Frame by frame animation

Zero level tutorial

Method:

(i) Draw any object.

(ii) Press "F6" on key board(F6 adds one key frame to your movie).

(iii) Make any change in your object(change its color or shape or any thing you like).

(iv) Your movie is completed , now test it by pressing "Ctrl+Enter" key on key board.

How to stop a movie

Double click last frame to open "frame Properties Dialogue Box".

Click "actions" tab.

Click "+" button.

Now click "stop" and press "OK"

In Flash an animated button consists of four frames as it has the following four states:

UP state: original shape of button or normal state of button.

OVER state: Change in the shape or color of button when the mouse rolls over the button.

DOWN state: Change in the shape or color of button when the user clicks on the button.

HIT state: we can't see this state in our movie. In hit state we draw an object that defines the boundary of button. It is very helpful when we use text as a button.

Create first Flash button

(i) Draw an object.

(ii) select object by arrow tool.

(iii) From "Insert menu" choose "convert to symbols" and behaviour as "button".

(iv) Right click button object and choose "Edit" from drop-down list.

(v) Press "F6" three times.

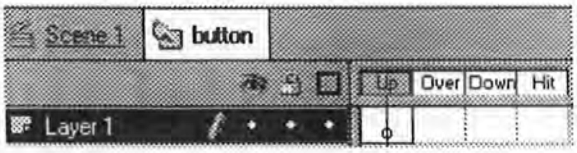
(vi) Click "over" states on the timeline and change the color of object

(vii) Click "Down" states on the timeline and change the color of object again.

(ix) Press "ctrl+Enter" to test movie.

Place your cursor over the button and click to view what is going on.

time line looks like that:



Mask effect

With the help of mask layers we can create animated effect in which an object in our movie hides and reveals with time. A mask layer can also produce spotlight effects . A mask layer hides everything on the layers linked to the mask layer. Mask layers can contain only a single shape.

Create a mask effect over text

(i)Type your name in layer 1.

(ii) Take fifty blank frames in layer 1(press F5 key 50 times).

(iii)lock layer1.

(iv) Add a new layer named "Layer 2".(note: layer 2 must be above the layer 1, if not drag it over layer 1)

(v) In layer 2 draw an oval on the left side of work area.

(vii) Select oval by arrow tool.

(viii) From "insert menu" choose "convert to symbol" and "graphic" as behavior.

(ix) Click frame 1 of layer 2, move oval to right by arrow tool (about 0.5 inch).

(x) Click frame 10 and press "F6" key of layer 2 .Now move oval to right (about 1.00 inch) by right arrow key on your keyboard.

(xi) Click frame 20 and press "F6" key of layer 2 .Now move oval to right (about 1.00 inch) by right arrow key on your keyboard.

(xii) Click frame 30 and press "F6" key of layer 2 .Now move oval to right (about 1.00 inch) by right arrow key on your keyboard.

(xiii) Click frame 40 and press "F6" key of layer 2 .Now move oval to right (about 1.00 inch) by right arrow key on your keyboard.

Click frame 50 and press "F6".

(xiv) Right click on each key frame 1, 10, 20, 30 and 40 one by one and choose "create motion tween".

(xv) Right click layer 2 and choose "mask".

(xvi) Test your movie.

Create motion along a path

(i) Create any object say circle.

(ii) Select circle by arrow tool.

(iii) Press F8 key on your keyboard.

(iv) In the dialogue box choose "graphic" as behavior.

(v) Click frame 50 and press "F6" key.

(vi) Double click frame 1.

(vii) Choose "motion" in tweening dialogue box.Click "OK".

Timeline looks like:

keep in mind that the circle has a "+" sign at the centre. (viii) add motion guide layer by clicking "motion guide" button at the bottom of layers as shown.

(x) Now click layer 1.

(xi) Select arrow tool.Drag circle to point 1(starting point of curved path).Match point 1 and "+" sign.

(xii) Click keyframe 50 in layer 1 and drag circle to point 2 and match "+" sign with the end point of path

(marked 2).

(xiii) Test movie.

Scale and rotate animation

In this type of animation size of object changes with rotation.

(i) draw any object

(ii) Select object by arrow tool.

(iii) From "insert menu" choose "convert to symbols".In dialogue box choose "graphic".

(iv) Click frame 25 and press "F6" key.

(v) Click frame 1.

(vi) From "modify menu" choose "transform" and then "scale and rotate".Enter values size=10 and rotate=280.Click "OK".

(vii) Right click on keyframe 1 and choose "create motion tween".
(viii) Now test your movie.

Create a shape tween animation

(i) Draw an oval in frame 1.

(ii) Click frame 30 and press "F6" key on your keyboard to insert a keyframe at frame 30.

(iii) Choose "arrow tool" and click out side the stage to deselect oval.

(iv) By using "arrow tool" change the shape of oval to any shape , change its color or whatever you wish.

(v) Double click frame 1.

(vi) Choose "tweening" tab from the dialogue box as described in previous lessons.

(vii) Choose "shape" from tweening drop down list and click "OK".

(viii) Now test movie by "Ctrl+Enter" key.

Chapter 5

Java script

What is java script?

Java Script is a compact, object-based scripting language for developing client and server Internet applications. JavaScript statements can be embedded directly in an HTML page. These statements can recognize and respond to user events such as mouse clicks, form input, and page navigation. For example, you can write a JavaScript function to verify that users enter valid information into a form. Without any network transmission, an HTML page with embedded JavaScript can interpret the entered text and alert the user with a message dialog if the input is invalid. Or you can use JavaScript to perform an action (such as play an audio file, execute an applet, or communicate with a plug-in) in response to the user opening or exiting a page.

Java Script

Is *a programmable API that allows cross-platform scripting of events, objects, and actions.* It allows the page designer to access events such as startups, exits, and users' mouse clicks. JavaScript extends the programmatic capabilities of most browsers to a wide range of authors, and is easy enough for anyone who can compose HTML.

Using JavaScript, even less-experienced developers will be able to direct responses from a variety of events, objects, and actions. It provides anyone who can compose HTML with the ability to change images and play different sounds in response to specified events, such as a users' mouse click or screen exit and entry.

Syntax and methodology of JavaScript

This JavaScript tutorial is aimed primarily at those who have had at least some exposure to another programming language. It is not our purpose here to cover the basic concepts of computer programming, but rather illustrate the syntax and methodology of JavaScript. For its part, JavaScript is a rather basic language which conforms tightly to the core concepts of computer programming. Any background in programming, from Visual Basic to Pascal to C (which is far more advanced) is sufficient to readily understanding JavaScript

JavaScript Grammar

JavaScript code, much like other programming languages, is made up of statements which serve to make assignments, compare values, and execute other sections of code. By and large, programmers will already be familiar with JavaScript's usage of variables, operators, and statements. Below is a chart summarizing the main elements of JavaScript grammar. Following, we will look at each element in detail.

Variables

Labels which refer to a changeable value.

Example: total may be possess a value of 100.

Operators

Actors which can be used to calculate or compare values.

Example: Two values may be summed using the addition operator(+);

total+tax

Example: Two values may be compared using the greater-than operator (>);

total>200

Expressions

Any combination of variables, operators, and statements which evaluate to some result. In English parlance this might be termed a "sentence" or even a "phrase", in that grammatical elements are combined into a cogent meaning. Example:

total=100;

Example: if (total>100)

Statements

As in English, a statement pulls all grammatical elements together into a full thought. JavaScript statements may take the form of conditionals, loops, or object manipulations. It is good form to separate statements by semicolons, although this is only mandatory if multiple statements reside on the same line.

Example: if (total>100) {statements;} else {statements;}

Example: while (clicks<10) {statements;}

Objects

Containing constructs which possess a set of values, each value reflected into an individual *property* of that object. Objects are a critical concept and feature of JavaScript. A single object may contain many properties, each property which acts like a variable reflecting a certain value. JavaScript can reference a large number of "built-in" objects which refer to characteristics of a Web document. For instance, the document object contains properties which reflect the background color of the current document, its title, and many more. For a fuller explanation of the built-in objects of JavaScript, see the section on "Document Object Model".

Functions and Methods

A JavaScript function is quite similar to a "procedure" or "subroutine" in other programming languages. A function is a discrete set of statements which perform some action. It may accept incoming values (parameters), and it may return an outgoing value. A function is "called" from a JavaScript statement to perform its duty. A method is simply a function which is contained in an object. For instance, a function which closes the current window, named close(), is part of the window object; thus, window.close() is known as a method

Variables and Data Types

Variables store and retrieve data, also known as "values". A variable can refer to a value which changes or is changed. Variables are referred to by name, although the name you give them must conform to certain rules. A JavaScript identifier, or name, must start with a letter or underscore ("_"); subsequent characters can also be digits (0-9). Because JavaScript is case sensitive, letters include the characters "A" through "Z" (uppercase) and the characters "a" through "z" (lowercase). Typically, variable names are chosen to be meaningful regarding the value they hold. For example, a good variable name for containing the total price of goods orders would be total.

scope

When you assign a new variable to an initial value, you must consider the issue of scope. A variable may be scoped as either global or local. A global variable may be accessed from any JavaScript on the page. A local variable may only be accessed from within the function in which it was assigned.

Commonly, you create a new global variable by simply assigning it a value:

```
newVariable=5;
```

However, if you are coding within a function and you want to create a local variable which only scopes within that function you must declare the new variable using the var statement:

```
function newFunction()
{ var loop=1;
 total=0;
 ...additional statements...
}
```

In the example above, the variable loop will be local to newFunction(), while total will be global to the entire page.

Type

A value, the data assigned to a variable, may consist of any sort of data. However, JavaScript considers data to fall into several possible *types*. Depending on the type of data, certain operations may or may not be able to be performed on the values. For example, you cannot arithmetically multiply two string values. Variables can be these types:

Numbers

3 or 7.987, Integer and floating-point numbers.

Integers can be positive, 0, or negative; Integers can be expressed in decimal (base 10), hexadecimal (base 16), and octal (base 8). A decimal integer literal consists of a sequence of digits without a leading 0 (zero). A leading 0 (zero) on an integer literal indicates it is in octal; a leading 0x (or 0X) indicates hexadecimal. Hexadecimal integers can include digits (0-9) and the letters a-f and A-F. Octal integers can include only the digits 0-7.

A floating-point number can contain either a decimal point, an "e" (uppercase or lowercase), which is used to represent "ten to the power of" in scientific notation, or both. The exponent part is an "e" or "E" followed by an integer, which can be signed (preceded by "+" or "-"). A floating-point literal must have at least one digit and either a decimal point or "e" (or "E").

Booleans

True or False. The possible Boolean values are true and false. These are special values, and are not usable as 1 and 0. In a comparison, any expression that evaluates to 0 is taken to be false, and any statement that evaluates to a number other than 0 is taken to be true.

Strings

"Hello World !" Strings are delineated by single or double quotation marks. (Use single quotes to type strings that contain quotation marks.)

Objects

myObj = new Object();

Null

Not the same as zero - no value at all. A null value is one that has no value and means nothing.

Undefined

A value that is undefined is a value held by a variable after it has been created, but before a value has been assigned to it.

That said, JavaScript is a loosely typed language -- you do not have to specify the data type of a variable when you declare it, and data types are converted automatically as needed during script execution. By and large, you may simply assign any type of data to any variable. The only time data typing matters is when you need to perform operations on the data. Certain operators behave differently depending on the type of data being deal with. For example, consider the + operator:

"5" + "10"	*yields*	"510" *(string concatenation)*
5 + 10	*yields*	15 *(arithmetic sum)*

Operators

Operators take one or more variables or values (*operands*) and return a new value; e.g. the '+' operator can add two numbers to produce a third. You use operators in expressions to relate values, whether to perform arithmetic or compare quantities. Operators are divided into several classes depending on the relation they perform:

Arithmetic or computational

Arithmetic operators take numerical values (either literals or variables) as their operands and return a single numerical value. The standard arithmetic operators are:

+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulus: the remainder after division; e.g. 10 % 3 yields 1.
++	Unary increment: this operator only takes one operand. The operand's value is increased by 1. The value returned depends on whether the ++ operator is placed before or after the operand; e.g. *++x* will return the value of *x* following the increment whereas *x++*

	will return the value of *x* prior to the increment.
--	Unary decrement: this operator only takes one operand. The operand's value is decreased by 1. The value returned depends on whether the -- operator is placed before or after the operand; e.g. --*x* will return the value of *x* following the decrement whereas *x*-- will return the value of *x* prior to the decrement.
-	Unary negation: returns the negation of operand.

Comparison

A comparison operator compares its operands and returns a logical value based on whether the comparison is true or not. The operands can be numerical or string values. When used on string values, the comparisons are based on the standard lexicographical (alphabetic) ordering.

==	"Equal to" returns true if operands are equal.
!=	"Not equal to" returns true if operands are not equal.
>	"Greater than" returns true if left operand is greater than right operand.
>=	"Greater than or equal to" returns true if left operand is greater than or equal to right operand.
<	"Less than" returns true if left operand is less than right operand.
<=	"Less than or equal to" returns true if left operand is less than or equal to right operand.

Boolean

Boolean operators are typically used to combine multiple comparisons into a conditional expression. For example, you might want to test whether (total>100) AND (stateTax=true). A boolean operator takes two operands, each of which is a true or false value, and returns a true or false result.

&&	"And" returns true if both operands are true.
||	"Or" returns true if either operand is true.
!	"Not" returns true if the negation of the operand is true (e.g. the operand is false).

String

Strings can be compared using the comparison operators. Additionally, you can concatenate strings using the + operator.

"dog" + "bert"	*yields*	"dogbert"

Assignment

The assignment operator (=) lets you assign a value to a variable. You can assign any value to a variable, including another variable (whose value will be assigned). Several shorthand assignment operators allow you to perform an operation and assign its result to a variable in one step.

=	Assigns the value of the righthand operand to the variable on the left. Example: total=100; Example: total=(price+tax+shipping)
+= (also -=, *=, /=)	Adds the value of the righthand operand to the lefthand variable and stores the result in the lefthand variable. Example: total+=shipping (adds value of *shipping* to *total* and assigned result to *total*)

&= (also |=)	Assigns result of (lefthand operand && righthand operand) to lefthand operand.

Special

Several JavaScript operators, rarely used, fall into no particular category. These operators are summarized below.

Conditional operator (condition) ? trueVal : falseVal	Assigns a specified value to a variable if a condition is true, otherwise assigns an alternate value if condition is false. Example: preferredPet = (cats > dogs) ? "felines" : "canines" If (cats>dogs), *preferredPet* will be assigned the string value "felines," otherwise it will be assigned "canines".
Type of *operand*	Returns the data type of *operand.* Example -- test a variable to determine if it contains a number: if (typeof total=="number") ...

Statements

Statements define the flow of a script, known as "program flow." A statement, like a fully grammatical English sentence, is made up of smaller expressions which, altogether, evaluate into a cogent meaning. In JavaScript, statements are organized as either conditionals, loops, object manipulations, and comments.

Good practice suggests that each JavaScript statements should be terminated with a semicolon (;). This is often not strictly necessary, as a new line also serves to separate statements, but when multiple statements reside on the same line the semicolon delimiter is mandatory.

A set of statements that is surrounded by braces is called a block. Blocks of statements are used, for example, in functions and conditionals.

Normally statements are executed sequentially: *x = 1; y = 2; z = x + y;* but this can be altered by some statements which test a condition and branch or loop according to the result.

Conditionals

Conditional statements direct program flow in specified directions depending upon the outcomes of specified conditions. These tests are a major influence on the order of execution in a program.

if...else

As seen in many programming languages, *if* the condition evaluates to true then the block of *statements1* is executed. Optionally, an *else* clause specifies a block of *statements2* which are executed otherwise. You may omit the else clause if there are no statements which need to be executed if the condition is false.

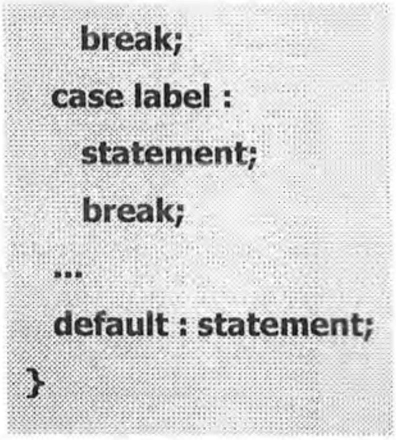
```
if (condition)
 { statements1; }

else
 { statements2; }
```

Switch (Netscape)

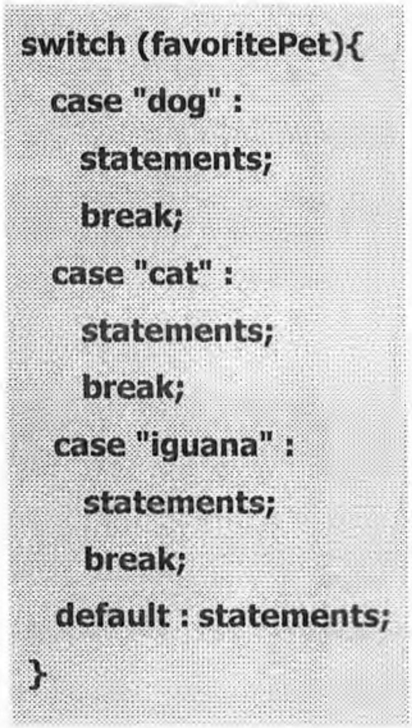
Commonly known as a "case statement," *switch* matches an expression with a specified case, and executes the statements defined for that case. In essence, the switch statement is a sort of shorthand for combining many implied *if* statements together.

```
switch (expression){
  case label :
    statement;
```



```
    break;
  case label :
    statement;
    break;
  ...
  default : statement;
}
```

For example, imagine that you wanted to execute different sets of statements depending on whether *favoritePet* was "dog," "cat," or "iguana." Note that the *break;* statement prevents any cases below the match from being executed. The *default* case is matched if none of the cases match the expression.



```
switch (favoritePet){
  case "dog" :
    statements;
    break;
  case "cat" :
    statements;
    break;
  case "iguana" :
    statements;
    break;
  default : statements;
}
```

Loops

for

The venerable *for* loop repeatedly cycles through a block of statements until a test condition is false. Typically, the number of times a loop is repeated depends on a counter. The JavaScript *for* syntax incorporates the counter and its increments:

```
for (initial-statement; test; increment)
 { statements; }
```

The initial-statement is executed first, and once only. Commonly, this statement is used to initialize a counter variable. Then the test is applied and if it succeeds then the statements are executed. The increment is applied to the counter variable and then the loop starts again. For instance, consider a loop which executes 10 times:

```
for (i=0; i<10; i++)
 { statements; }
```

do...while (Netscape)

Another loop, a *do...while* statement executes a block of statements repeatedly until a condition becomes false. Due to its structure, this loop necessarily executes the statement at least once.

```
do
 { statements;}
while (condition)
```

while

In similar fashion as the *do...while* statement, the *while* statement executes its statement block as long as the condition is true. The main difference between *while* and *do...while*, aside from the fact that only *while* is supported in all JavaScript versions, is that a while loop may not execute the statements even once if the condition is initially false.

```
while (condition)
 { statements; }
```

break and continue

Both of these statements may be used to "jump the tracks" of an iterating loop. When used within the statement block of a loop, each statement behaves slightly differently:

Break	Aborts execution of the loop, drops out of loop to the next statement following the loop.
Continue	Aborts *this single* iteration of the loop, returns execution to the loop control, meaning the condition specified by the loop statement. Loop may execute again if condition is still true.

Object manipulation

for...in

The sometimes confusing *for...in* statement is used to cycle through each property of an object or each element of an array. The idea is that you may want to execute a statement block which operates on every property or element.

```
for (variable in object)
 { statements; }
```

Imagine, for example, that an object named *wine1* has five properties: vineyard, year, varietal, alcohol, and color. You want to output the value of each property, as if producing a record from a database.

```
var record = "Wine 1<br><br>"
for (var prop in wine1)
 {record += prop + " = " + wine1[prop] + "<BR>"}
record += "<br>"
document.write(record)
```

with

The *with* statement serves as a sort of shorthand, allowing you to execute a series of statement who all assume a specified object as the reference. In other words, the object specified in the *with* statement is used as the default object whenever a property is encountered with no object specified.

```
with (object)
 { statements; }
```

Comments

Despite the fact that comments are purely optional, they can easily be a crucial part of your program. Comments can explain the action, like a color commentary, which can be a great help in understanding the code. Whether as a teaching tool or to simply remind yourself what the code does, comments are best sprinkled liberally throughout a program. Remember, comments are for humans, so write them that way!

Comments can also be used for debugging -- you can comment "out" sections of code to prevent them from being executed. In doing so you may learn more about why a certain problem is occurring in your program.

Because JavaScript must ignore comments, there is an appropriate syntax for demarcating text as a comment. For single line comments, simply precede the line with two backslashes. For multi-line comment blocks, begin the comment with /* and close with */.

```
//A lonely ol' single line comment
/* A dense thicket of commentary, spanning many captivating lines
of explanation and intrigue. */
```

Functions

A function groups together a set of statements under a named subroutine. This allows you to conveniently "call" the function whenever its action is required. Functions are a fundamental building block of most JavaScript programs, so you'll become quite familiar with their use. Before you can call on a function, of course, you must first create it. We can break down the use of functions, then, into two logical categories: defining functions and calling functions.

Defining functions

The function definition is a statement which describes the function: its name, any values (known as "arguments") which it accepts incoming, and the statements of which the function is comprised.

```
function funcName(argument1,argument2,etc)
{ statements; }
```

A function doesn't necessarily require arguments, in which case you need only write out the parenthesis; e.g. funcName(). If you do specify arguments, those arguments will be variables within the function body (the statements which make up the function). The initial values of those variables will be any values passed on by the function call.

Generally it's best to define the functions for a page in the HEAD portion of a document. Since the HEAD is loaded first, this guarantees that functions are loaded before the user has a chance to do anything that might call a function. Alternately, some programmers place all of their functions into a separate file, and include them in a page using the SRC attribute of the SCRIPT tag. Either way, the key is to load the function definitions before any code is executed. Consider, for example, a simple function which outputs an argument to the Web page, as a bold and blinking message:

```
function boldblink(message)
{
document.write("<blink><strong>"+message+"</strong></blink>"
); }
```

Some functions may return a value to the calling expression. The following function accepts two arguments, x and y, and returns the result of x raised to the y power:

```
function raiseP(x,y)
{ total=1;
 for (j=0; j<y; j++)
  { total*=x; }
 return total; //result of x raised to y power
}
```

calling functions

A function waits in the wings until it is called onto the stage. You call a function simply by specifying its name followed by a parenthetical list of arguments, if any:

```
clearPage();
boldblink("Call me gaudy!");
```

Functions which return a result should be called from within an expression:

```
total=raiseP(2,8);
if (raiseP(tax,2)<100) ...
```

Objects

An object is a "package" of data; a collection of properties (variables) and methods (functions) all classed under a single name. For example, imagine that there was an object named *car*. We could say that the *car* object possesses several properties: make, model, year, and color, for example. We might even say that car possesses some methods: go(), stop(), and reverse(). Although *car* is obviously fictional, you can see that its properties and methods all relate to a common theme.

In JavaScript you may create your own objects for storing data. More commonly, though, you will use the many "built-in" objects which allow you to work with, manipulate, and access the Web page and Web browser. This set of pre-existing objects is known as the "Document Object Model".

Properties

Access the properties of an object with a simple notation: objectName.propertyName. Both the object name and property name are case sensitive, so watch your typing. Because a property is essentially a variable, you can create new properties by simply assigning it a value. Assuming, for instance, that carObj already exists (we'll learn to create a new object shortly), you can give it properties named make, model, and year as follows:

```
carObj.make="Toyota";
carObj.model="Camry";
carObj.year=1990;
document.write(carObj.year);
```

A JavaScript object, basically, is an array. If you're familiar with other languages you probably recognize an array as a collection of values residing within a single named data structure. You can access an object's properties either using the objectName.propertyName syntax illustrated above, or by using an array syntax:

```
carObj["make"]="Toyota";
carObj["model"]="Camry";
document.write(carObj["year"]);
```

Methods

Unlike a basic data array, an object can also contain functions, which are known as *methods* when part of an object. You call a method using the basic syntax: objectName.methodName(). Any arguments required for the method are passed between the parentheses, just like a normal function call.

For example, the window object possesses a method named *close()*, which simply closes the specified browser window:

```
window.close();
```

Creating Objects

Most of the time you will be referencing objects which are built-in to the DOM. However, you may want to create your own objects for storing data within a JavaScript program. There are several ways to create a new object, but we'll look at two: creating a direct instance of an object and creating an object prototype.

direct instance of an object

Despite the awkward sound name, a "direct instance of an object" simply means creating a new single object, such as *myPetDog*:

```
myPetDog=new Object();
 myPetDog.name="Barney";
 myPetDog.breed="beagle";
 myPetDog.year=1981;
```

Assigning a method to your new object is also simple. Assume that you already have coded a function named *woof()*, which causes a barking sound to play:

```
myPetDog.woof=woof;
```

prototype of an object

Sometimes, you'll want to create a "template" or prototype of an object. This does not create an actual instance of the object, but defines the structure of the object. In the future, then, you can quickly stamp out a particular instance of the object. Suppose that instead of *myPetDog*, you created a prototype object named *petDog*. This object could then be a template for a particular pet dog object. First, create a function which defines the *petDog* structure:

```
function petDog(name, breed, year)
{ this.name = name;
  this.breed = breed;
  this.year = year;
}
```

Now that the *petDog* prototype has been set, you can quickly create single instances of a new object based on the *petDog* structure:

```
myPetDog=new petDog("barney","beagle",1981);
yourPetDog=new petDog("max","terrier",1990);
```

Conclusion

Like an essay in English, a JavaScript program is a series of statements which work together towards a particular goal, and are made up of component grammatical elements, such as expressions and operators. Because JavaScript is a programming language invented "for the Web," it is oriented towards the specific needs of Web developers. The set of pre-built objects largely reflect characteristics of the Web page, allowing your JavaScript program to manipulate, modify, and react to the Web page.

Interactivity is the driving force behind JavaScript, and so most JavaScript programs are launched by actions which occur on the Web page, often by the user. In doing so, JavaScript's purpose is to nudge Web pages away from static displays of data towards applications which can process and react.

Introduction to HTML

What is html?

HTML is composed of tags. HTML tags are always enclosed in angle-brackets (< >) and are case-insensitive; that is, it doesn't matter whether you type them in upper or lower case. I almost always use upper case, because it makes the tags easier to pick out in a document, but that's just me. You can do whatever you like.

Tags typically occur in begin-end pairs. These pairs are in the form

  <tag> ... </tag>

where the <tag> indicates the beginning of a tag-pair, and the </tag> indicates the end. (The three dots indicate an arbitrary amount of content between the tags.) The usual way to refer to each tag is "tag" for the first and "slash-tag" for the second, where tag is the actual name of the tag being discussed.

These pairs define containers. Any content within a container has the rules of that container applied to it. For example, the text within a "boldface container" would be boldfaced. Similarly, paragraphs are defined using a "paragraph container."

Thinking of tag-sets as containers will help in another way: it will help you remember that tags should always be balanced. In other words, you should keep containers nested within each other, just as you would have to do in the real world. Let's try some visual examples where we actually draw the containers:

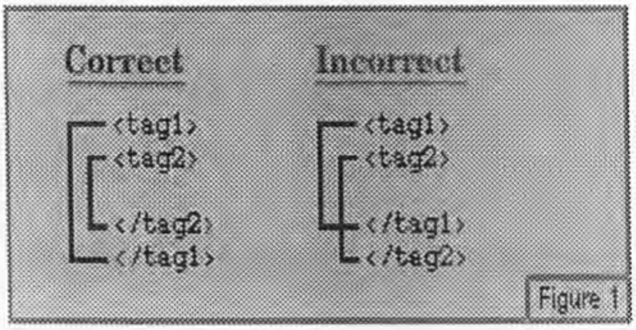


Figure 1

Why should you worry about this? Well, if you start overlapping containers as shown on the right, about the best you can expect is that the document will be formatted in unexpected ways.

One more thing to keep in mind with regards to containers. Since HTML is based on these structures, it is often the case that the arrangement of text within a container is irrelevant. For example, within a paragraph container, all of the text can be in one long line, or in a series of separate lines, or with every word on its own line, or with every word separated from every other by nineteen spaces. These would all be displayed exactly the same.

Therefore, try to keep in mind this thought: white space doesn't matter. (White space is all of the blank areas in a text file--empty lines, extra spaces, and so on.) I'll mention this again when discussing the paragraph tag, and it will crop up in other places. Again: white space doesn't matter.

Having said all that, I will now attempt to muddy the waters a bit by mentioning that not every tag in HTML is paired. Some tags, such as the line-break tag, stand on their own (that is, they have no closing tag). These are known as empty tags. As we encounter them, I'll point them out.

Document Tags

By "document tags," I mean the tags which divide up a Web page into its basic sections, such as the header information and the part of the page which contains the displayed text and graphics

HTML

The first and last tags in a document should always be the HTML tags. These are the tags that tell a Web browser where the HTML in your document begins and ends. The absolute most basic of all possible Web documents is:

```
<HTML>
</HTML>
```

HEAD

The HEAD tags contain all of the document's header information. When I say "header," I don't mean what appears at the top of the browser window, but things like the document title and so on. Speaking of which...

TITLE

This container is placed within the HEAD structure. Between the TITLE tags, you should have the title of your document. This will appear at the top of the browser's title bar, and also appears in the history list. Finally, the contents of the TITLE container go into your bookmark file, if you create a bookmark to a page.

BODY

BODY comes after the HEAD structure. Between the BODY tags, you find all of the stuff that gets displayed in the browser window. All of the text, the graphics, and links, and so on -- these things occur between the BODY tags.

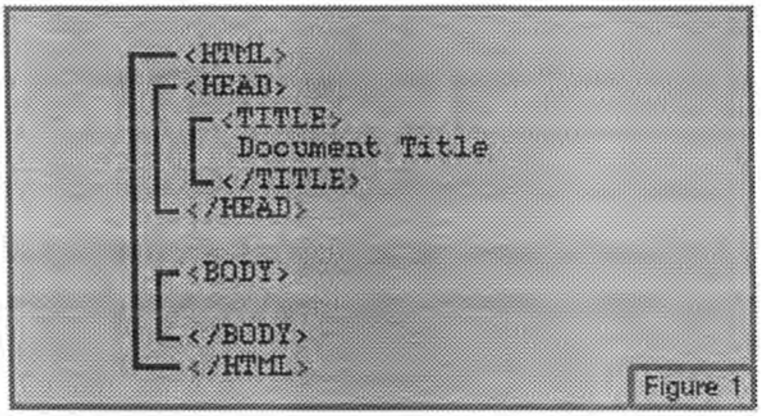
So, putting everything we've covered thus far into one file, we have:

```
<HTML>
<HEAD>
 <TITLE>Document Title</TITLE>
</HEAD>

<BODY>


</BODY>
</HTML>
```

This time, the result would be a document with a completely blank browser window, but at least the words "Document Title" would appear in the browser's history list



```
<HTML>
<HEAD>
  <TITLE>
   Document Title
  </TITLE>
</HEAD>

<BODY>

</BODY>
</HTML>
```

Figure 1

Comment Tags

If you want to leave yourself notes in an HTML document, but don't want those notes to show up in the browser window, you need to use the comment tag. To do that, you would do the following:

<!-- Hi, I'm a comment. -->

Your note would go where the text Hi, I'm a comment. appears. Yes, you do need an exclamation point after the opening bracket, but not before the closing bracket. That's the way the standard is written. I have no idea why. Also, there is no end tag; that is, a tag like </!-- text --> does not exist. The comment tag is not a container. This is our first example of an empty tag.

Basic Text Structures

Headings

There are six levels of headings, from Heading 1 through Heading 6. Heading 1 (H1) is "most important" and Heading 6 (H6) is "least important." By default, browsers will display the six heading levels in the same font, with the point size decreasing as the importance of the heading decreases. Here are all six HTML pairs, in descending order of importance:

```
<H1>Heading 1</H1>
<H2>Heading 2</H2>
<H3>Heading 3</H3>
<H4>Heading 4</H4>
<H5>Heading 5</H5>
<H6>Heading 6</H6>
```

These six lines, when placed into an HTML document, will simply display the six levels of headings.

Paragraphs

The beginning of a paragraph is marked by <P>, and the end by </P>]

Line Break

So what if you **want** to end a line after a certain word, but don't want to start a new paragraph? Well, what you need is a line break, which is invoked by using the <BR> tag. This forces a line break wherever you place it in the content (that is, whatever is after the <BR> tag will start from the left margin of the next line on the screen.)

And no, there is no </BR> tag. The line break tag is -- that's right! -- an empty tag. And when you think about it, this makes sense. The concept of a line break beginning and ending doesn't really work, since a line break is a one-shot occurrence.

Block quote

Block quotes are handy for those long pieces of text, which are quoted material and therefore need to be set apart and indented. That is exactly what block quote does. For example:

The block quote tags surround this section of text. A block quote can exist inside of a paragraph, and always lives on its own line (which is to say, there is an implied line break before and after the block quote, just as with headings or paragraphs themselves).

Block quotes are set up as follows:

<Block quote> ...*text*... </block quote>

Just like most other things in HTML, it's a container.

Lists

While simple in concept, lists can be very powerful in execution. There are three types of lists: unordered lists, ordered lists, and definition lists. The first two are very similar in structure, while definition lists have a unique setup.

Unordered Lists

The term "unordered list" may be a bit unfamiliar to you, but odds are you've heard of the "bullet list." That's exactly what an unordered list is -- a list of items, each one preceded by a "bullet" (a distinctive character; typically, a small black circle).

The list begins and ends with the tags <UL> and </UL> respectively. Each item in the list is marked using the <LI> tag, which stands for "List Item." <LI> has a corresponding </LI>, but this closing tag is not required to end the list item (although you could use one if you really wanted to). You can use as many list items as you like, up to your browser's built-in maximum, if any.

Here's the markup for a simple list:

```
<UL>
<LI>Monday
<LI>Tuesday
<LI>Wednesday
<LI>Thursday
<LI>Friday
</UL>
```

If you loaded an HTML page containing the markup above, you would see the days of the week, each one preceded by a "bullet." To wit:

Monday

Tuesday

Wednesday

Thursday

Friday

Almost anything can be put into a list item -- line breaks, entire paragraphs, images, links, or even other lists. For example:

```
<UL>
<LI>Monday
<LI>Tuesday
<LI>Wednesday
 <UL>
 <LI>6am - 9am
 <LI>9am - 12n
 <LI>12n - 3pm
 <LI>3pm - 6pm
 </UL>
<LI>Thursday
<LI>Friday
</UL>
```

In the above case, under "Wednesday" in the 'outer list,' you would find another unordered list (the three-hour blocks of time), which is referred to as a *nested list.* (In the markup above, I have indented the nested list for purposes of clarity; this is not required for the lists to work. Remember what I've said about whitespace...) Here's how it looks:

Monday

Tuesday

Wednesday

6am - 9am

9am - 12n

12n - 3pm

3pm - 6pm

Thursday

Friday

In theory, you could probably nest lists indefinitely, but a bit of restraint is called for. Don't nest them too deeply unless you absolutely have to, if for no other reason than aesthetics.

Ordered Lists

On the face of it, ordered lists look a lot like unordered lists, and a lot of the same rules apply to both constructs. The only difference in HTML is that instead of using <UL> and </UL>, an ordered list is contained within the tags <OL> and </OL>. Ordered lists are based on list items, just as unordered lists are.

However, when an ordered list is displayed in a Web browser, it uses an automatically generated sequence of item markers. In other words, the items are numbered. The markup for a simple ordered list, based on the first example in this chapter:

```
<OL>
<LI>Monday
<LI>Tuesday
<LI>Wednesday
<LI>Thursday
<LI>Friday
</OL>
```

The above markup will look similar to the previously discussed simple unordered list, with the important difference that each day of the week is numbered instead of preceded by a "bullet." In other words, it looks like this:

Monday

Tuesday

Wednesday

Thursday

Friday

Ordered lists are as nestable as unordered lists, and you can nest unordered lists in ordered lists, as well as the other way around.

Definition Lists

As you might expect, definition lists begin and end with the tags <DL> and </DL>. However, unlike the unordered and ordered lists, definition lists are **not** based on list items. They are instead based on term-definition pairs.

Here's the markup for a basic definition list:

```
<DL>
<DT>Do
<DD>a deer, a female deer
<DT>Re
<DD>a drop of golden sun
<DT>Mi
<DD>a name I call myself
<DT>Fa
<DD>a long, long way to run
<DT>Sol
<DD>a needle pulling thread
<DT>La
<DD>a note to follow so
<DT>Ti
<DD>a drink with jam and bread
</DL>
```

A good way to think of it is that <DT> stands for "**D**efinition-list **T**erm" and <DD> stands for "**D**efinition-list **D**efinition." When the above list is displayed, it arranges the elements such that each term is associated with the corresponding definition. The exact arrangement of elements may vary from browser to browser. Here's how the above markup comes out:

Do

a deer, a female deer

Re

a drop of golden sun

Mi

a name I call myself

Fa

a long, long way to run

Sol

a needle pulling thread

La

a note to follow so

Ti

a drink with jam and bread

Similar to ordered and unordered lists, definition lists can be arbitrarily long. Almost any structure can be placed in a <DD> tag, but putting large-scale structures (such as paragraphs, headings, and other lists) in the <DT> tag is not legal, according to the HTML Specification 2.0. You can leave out one part of a DT-DD pair, but this is not recommended.

Definition lists are perfect for creating glossaries.

There is one attribute to the <DL> tag, which is compact. This causes the display of the definition list to be compacted. What does that mean? It means that the information contained in the <DD> will be displayed on the same line as the <DT> term, if possible. (By the way, Microsoft's Internet Explorer does not support this attribute, so the examples in this section aren't going to work if you're using Explorer.) The markup would start out:

```
<DL compact>
<DT>Do
<DD>a deer, a female deer
<DT>Re
```

```
<DD>a drop of golden sun
.....
```

...and the entire compacted list would look a bit different than the first definition-list example. Thus:

Do

a deer, a female deer

Re

a drop of golden sun

Mi

a name I call myself

Fa

a long, long way to run

Sol

a needle pulling thread

La

a note to follow so

Ti

a drink with jam and bread

Special Effect Tags

Logical Style Tags

The "correct" way to highlight text is to use the logical tags, which do not directly specify the type of highlighting they will employ. There are 'defaults' written into the specification (see the quotations below) but there is no direct rule about which tag should be displayed in what way. This is entirely in keeping with HTML's structural nature.

Emphasis

To quote from the HTML 2.0 specification, the <EM> and </EM> tags provide "typographic emphasis, typically italics."

Strong

Again from the specification, the <STRONG> and </STRONG> tags provide "strong typographic emphasis, typically bold."

Citation

<CITE> and </CITE> specify a citation; this includes information like book titles, references, and so on. The text is usually displayed using italics.

These tags are recommended for use because they leave the most control to the reader of a document. However, in the real world, the tags in the next section are a lot more popular than those above.

Forced Style Tags

The tags I will cover here are sometimes called *forced style tags,* because their very nature forces a certain style within the document (at least, that's the idea). This does run counter to the entire "HTML is purely structural" philosophy, but my advice is not to worry about it too much. As long as you use the logical style tags where appropriate, then you're fine.

The four most commonly used forced style tags are very simple:

Boldface

Everything between <B> and </B> is **boldfaced**.

Italics

Everything between <I> and </I> is *italicized.*

Underline

Everything between <U> and </U> should be underlined; however, see the note below.

Typewriter Text

Everything between <TT> and </TT> is in typewriter text (a monospaced font in most browsers). This is typically used for variable names, or to show snippets of HTML.

Anchors

The Basic Anchor

The simplest possible anchor starts with <A> and ends with </A>. However, you will never ever use the <A> tag by itself, because it doesn't do anything. You'll need to enhance the <A> tag with attributes like...

HREF

HREF stands for "**H**ypertext **REF**erence," which is another way of saying, "The location of the file I want to load." Most anchors are in the form <A HREF="my doc\main page ">, where picture.gif is the location of the resource to which you want the link to point. For example

Check out the <A HREF=" my doc\main page">main page </A>--
it's pretty cool!

NAME

Using the NAME attribute, you can invisibly mark certain points of a document as places that can be jumped to directly, instead of loading the document and then scrolling around to find what you're after. This is accomplished by using a *named anchor*, which is slightly different than the anchor used to create a hyperlink.

Setting a named anchor is done using the form <A NAME="*label*"> ... </A>, where *label* is any text you care to use. It could be anything from chapter1 to 2.4.1 to oscar-the-grouch. So putting a name of pt.3 to the text "Part 3: Bagels and You" would be accomplished like this...

```
<A NAME="pt.3">Part 3: Bagels and You</A>
```

...and would look like this:

Part 3: Bagels and You

Note that there is no obvious, visible way to tell that the text has been named. This is as it should be. The only way named anchors are important is if they're referred to somewhere else. Also note that the HREF attribute does not appear in this anchor. It can do so, but it is not required; the only requirement is that an anchor have either an HREF or a NAME attribute.

Images

In order to make the IMG tag work, you need to use an SRC attribute. SRC stands for "source," as in, "the source of this graphic." (One way to read a typical image tag is "image source equals..." You'll see what I mean in a minute.) The value of SRC is the URL of the graphic you want to have displayed on your Web page. Thus, a typical image tag will take the form:

<IMG SRC="*URL of graphic*">

Chapter 6

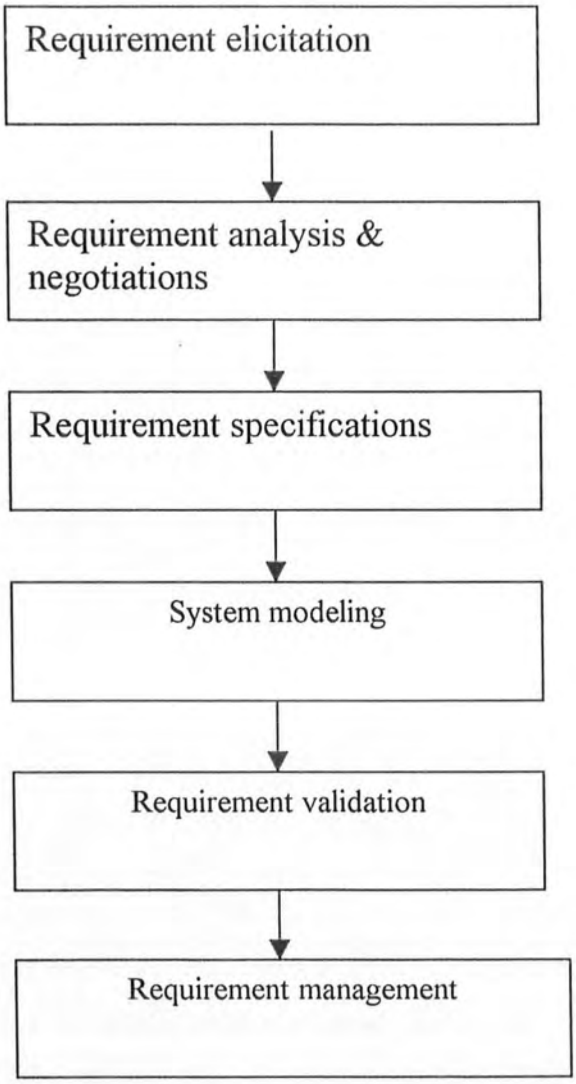
Software scheduling

The process adopted for carrying out the whole schedule of the project was more or the same as demanded by the software engineering methodology. we followed the following steps in brief:

1. Identifying the software requirements
2. Analyzing the requirements through negotiations
3. the role of the people , hardware ,software, data base , procedures & other system elements must be identified
4. operational requirements must be elicited , and analyzed, specified, modeled validated & managed

These activities are the foundations of software engineering.

BLOCK DIAGRAM SHOWING THE FLOW OF SOFTWARE PROCESS

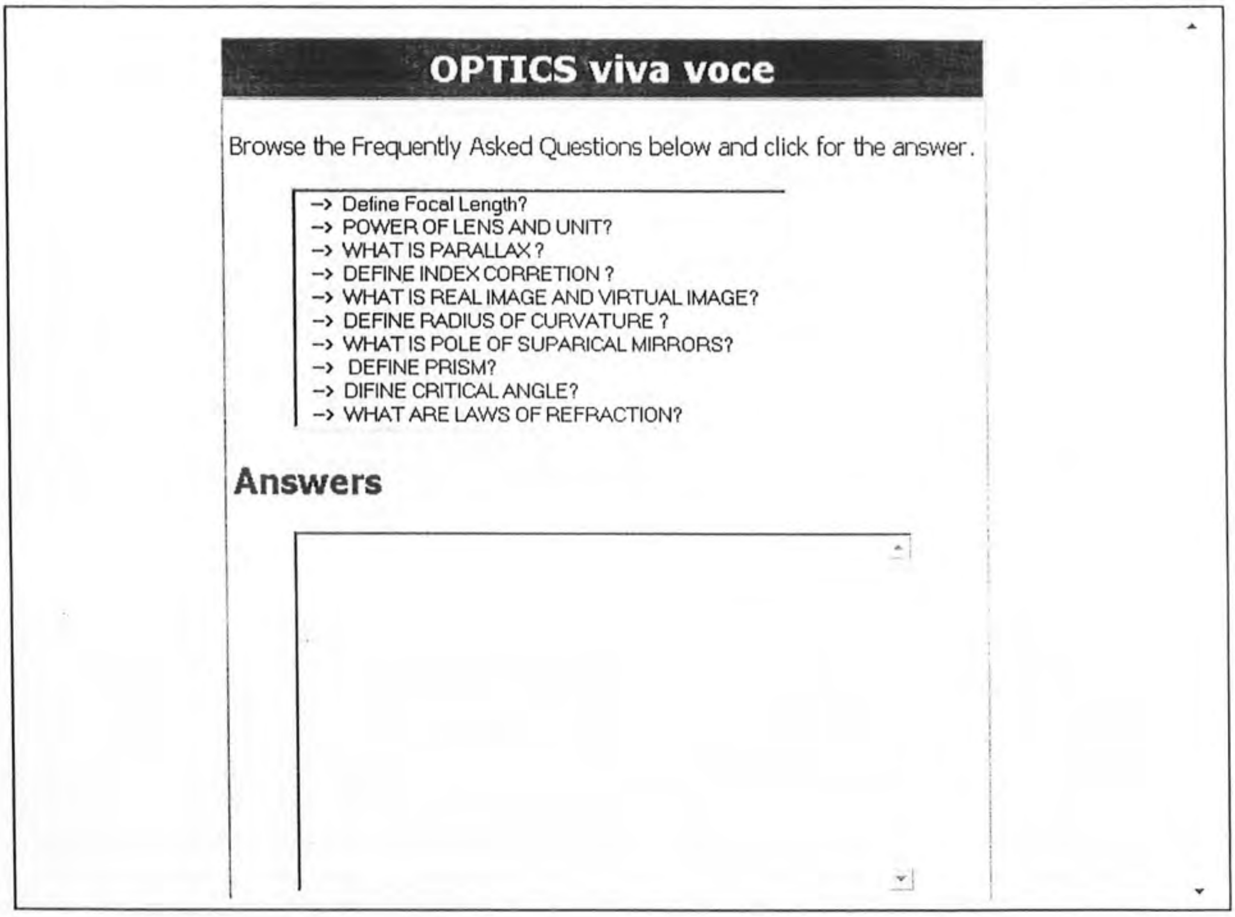


RESULTS

OPTICS (VIVA VOCE)

How to use this page.

Click on the question and see the its answer in the text box

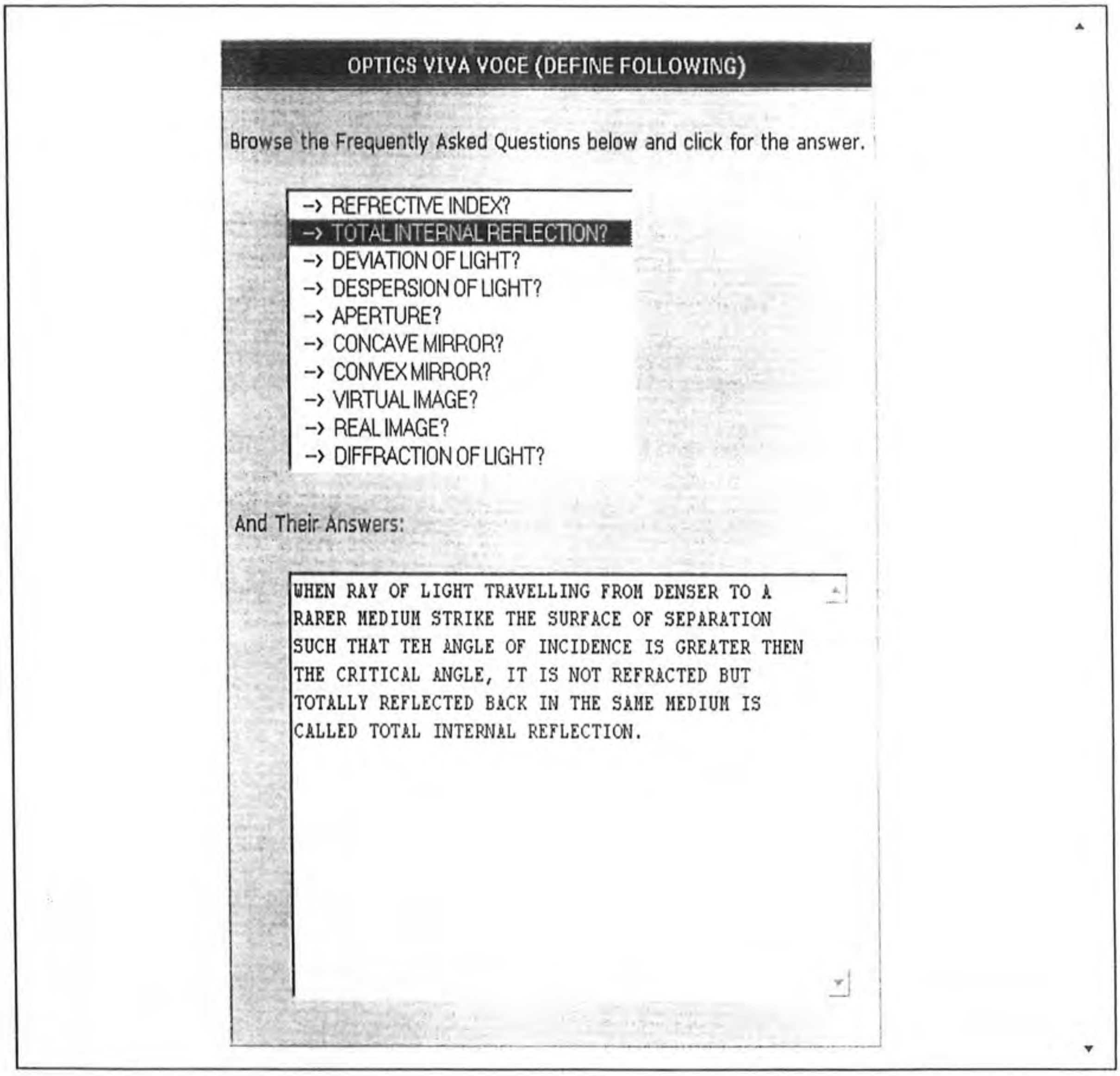


OPTICS viva voce

Browse the Frequently Asked Questions below and click for the answer.

--> Define Focal Length?
--> POWER OF LENS AND UNIT?
--> WHAT IS PARALLAX ?
--> DEFINE INDEX CORRETION ?
--> WHAT IS REAL IMAGE AND VIRTUAL IMAGE?
--> DEFINE RADIUS OF CURVATURE ?
--> WHAT IS POLE OF SUPARICAL MIRRORS?
--> DEFINE PRISM?
--> DIFINE CRITICAL ANGLE?
--> WHAT ARE LAWS OF REFRACTION?

Answers



OPTICS VIVA VOCE (DEFINE FOLLOWING)

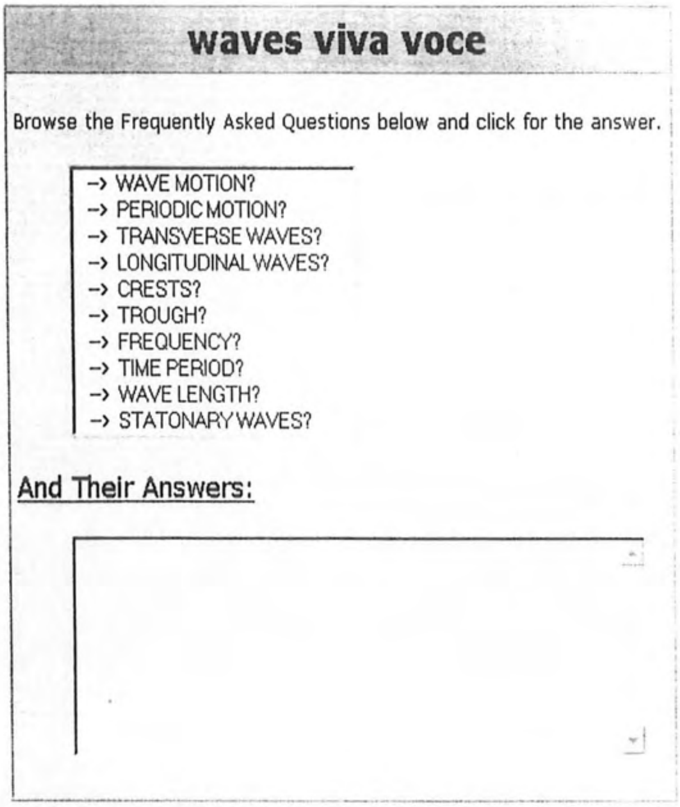
Browse the Frequently Asked Questions below and click for the answer.

-> REFRECTIVE INDEX?
-> TOTAL INTERNAL REFLECTION?
-> DEVIATION OF LIGHT?
-> DESPERSION OF LIGHT?
-> APERTURE?
-> CONCAVE MIRROR?
-> CONVEX MIRROR?
-> VIRTUAL IMAGE?
-> REAL IMAGE?
-> DIFFRACTION OF LIGHT?

And Their Answers:

WHEN RAY OF LIGHT TRAVELLING FROM DENSER TO A RARER MEDIUM STRIKE THE SURFACE OF SEPARATION SUCH THAT TEH ANGLE OF INCIDENCE IS GREATER THEN THE CRITICAL ANGLE, IT IS NOT REFRACTED BUT TOTALLY REFLECTED BACK IN THE SAME MEDIUM IS CALLED TOTAL INTERNAL REFLECTION.

WAVES

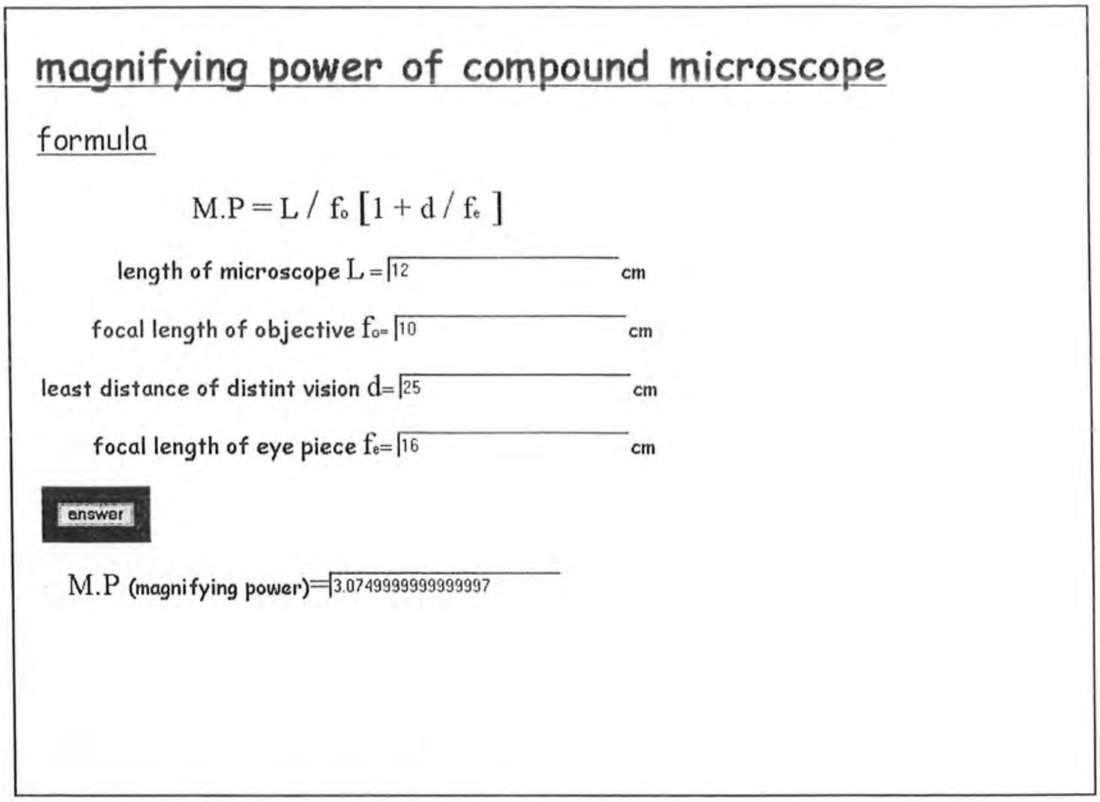


waves viva voce

Browse the Frequently Asked Questions below and click for the answer.

-> WAVE MOTION?
-> PERIODIC MOTION?
-> TRANSVERSE WAVES?
-> LONGITUDINAL WAVES?
-> CRESTS?
-> TROUGH?
-> FREQUENCY?
-> TIME PERIOD?
-> WAVE LENGTH?
-> STATONARY WAVES?

And Their Answers:



magnifying power of compound microscope

formula

$$M.P = L / f_o \left[1 + d / f_e\right]$$

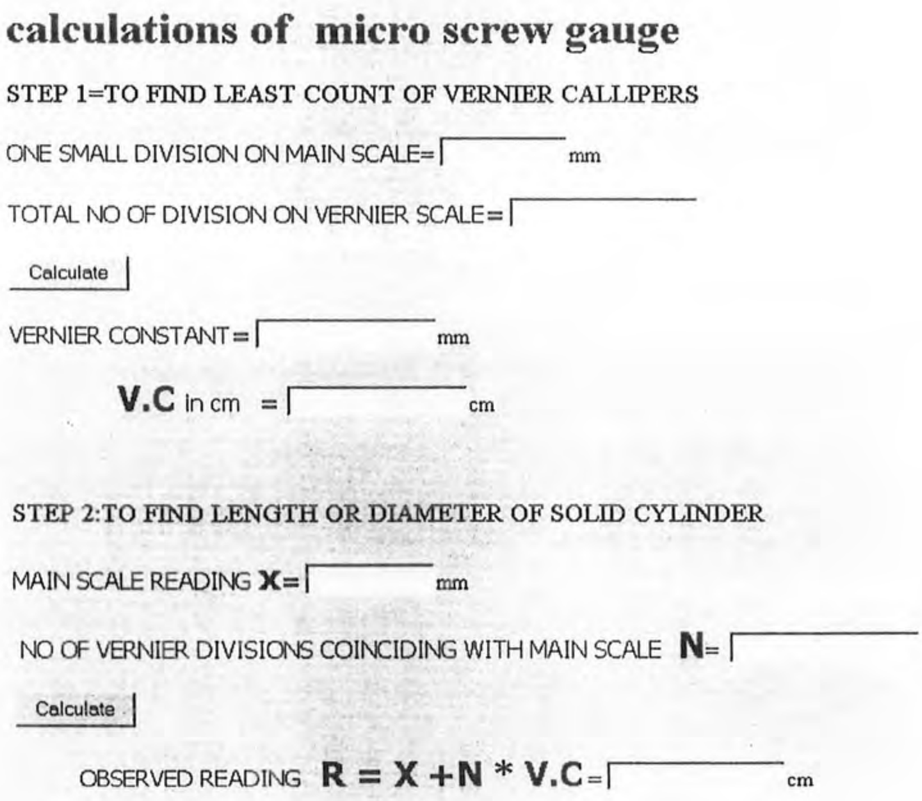
length of microscope $L =$ 12 cm

focal length of objective $f_{o}=$ 10 cm

least distance of distint vision $d=$ 25 cm

focal length of eye piece $f_e=$ 16 cm

M.P (magnifying power)= 3.0749999999999997



calculations of micro screw gauge

STEP 1=TO FIND LEAST COUNT OF VERNIER CALLIPERS

ONE SMALL DIVISION ON MAIN SCALE= mm

TOTAL NO OF DIVISION ON VERNIER SCALE=

VERNIER CONSTANT= mm

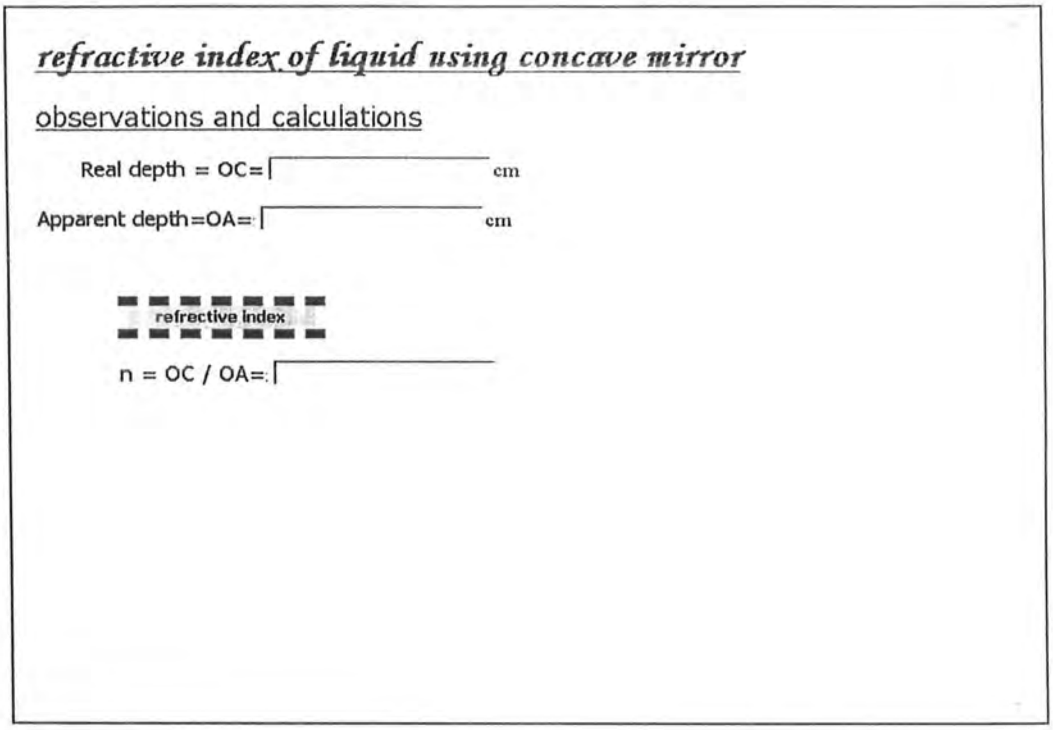
V.C in cm = cm

STEP 2:TO FIND LENGTH OR DIAMETER OF SOLID CYLINDER

MAIN SCALE READING **X=** mm

NO OF VERNIER DIVISIONS COINCIDING WITH MAIN SCALE **N**=

OBSERVED READING **R = X +N * V.C**= cm



refractive index of liquid using concave mirror

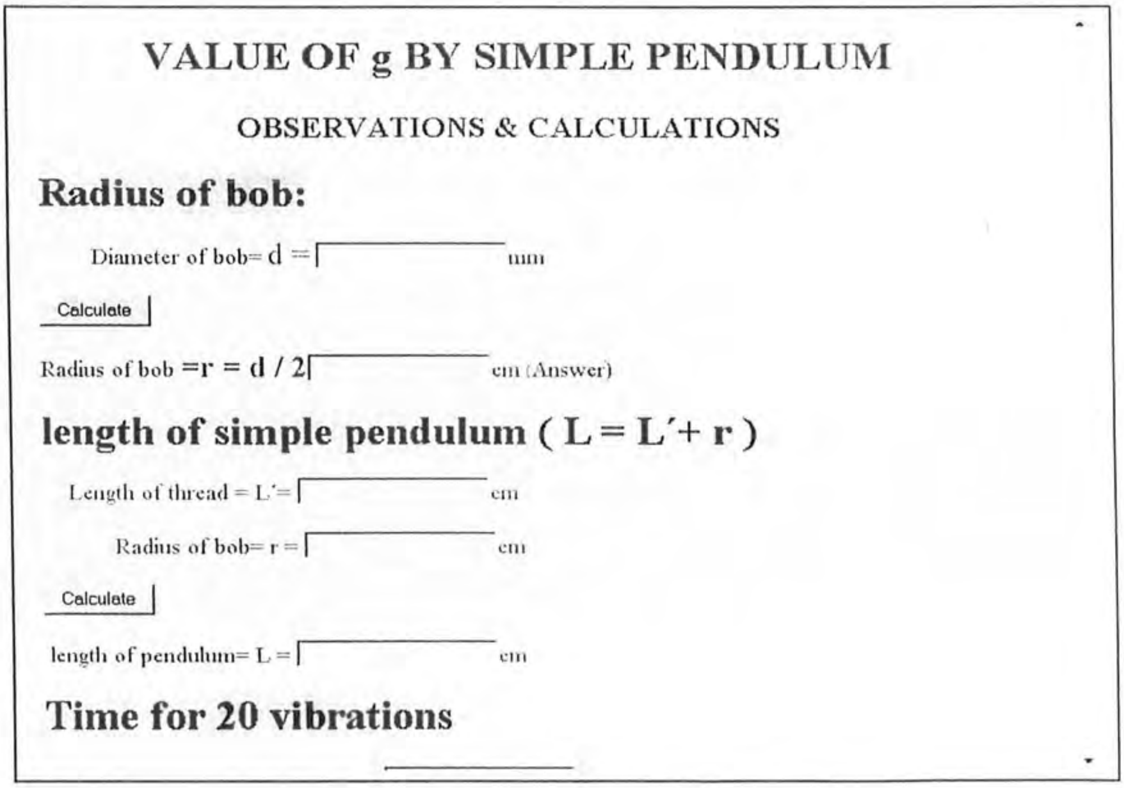
observations and calculations

Real depth = OC= cm

Apparent depth=OA= cm

refrective index

n = OC / OA=



VALUE OF g BY SIMPLE PENDULUM

OBSERVATIONS & CALCULATIONS

Radius of bob:

Diameter of bob= d = mm

Calculate

Radius of bob =r = d / 2 cm (Answer)

length of simple pendulum (L = L′+ r)

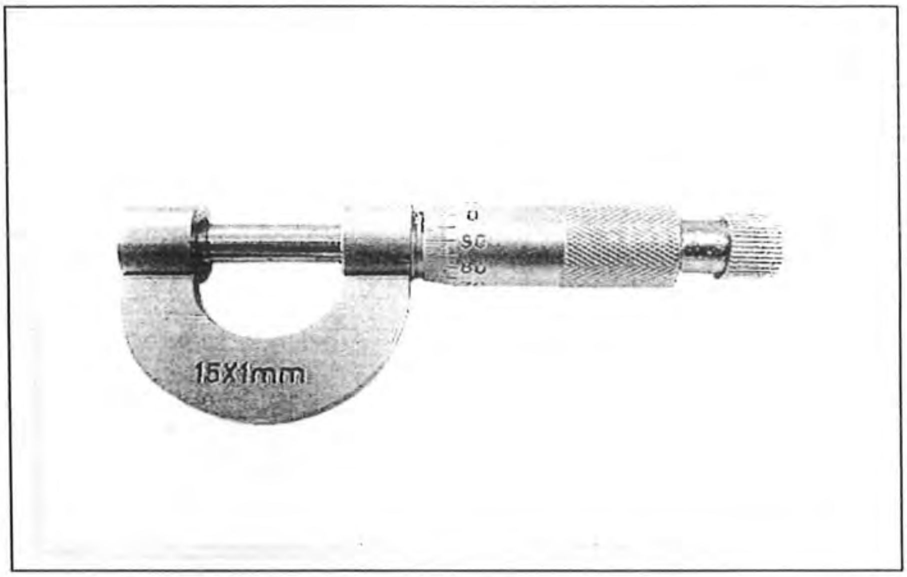
Length of thread = L′= cm

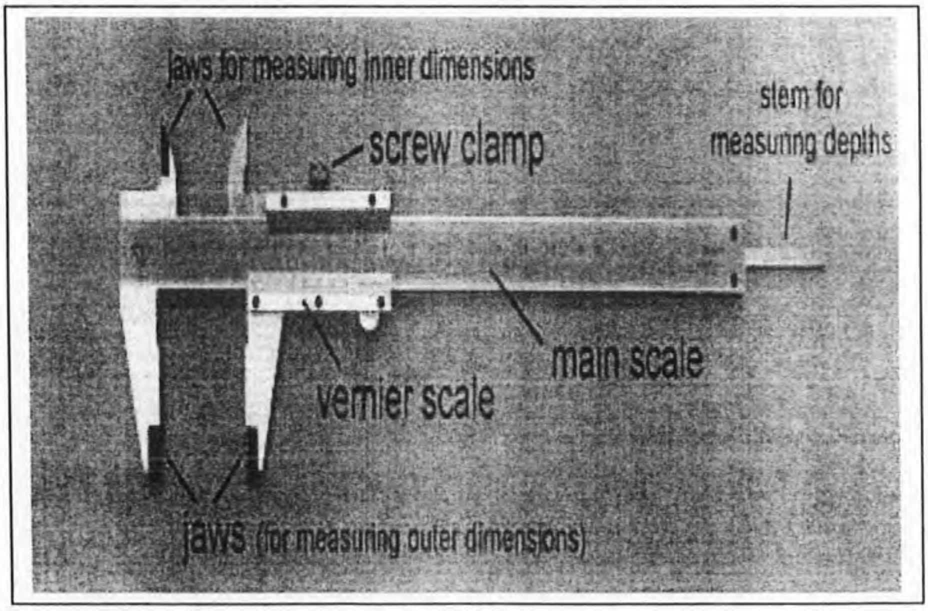
Radius of bob= r = cm

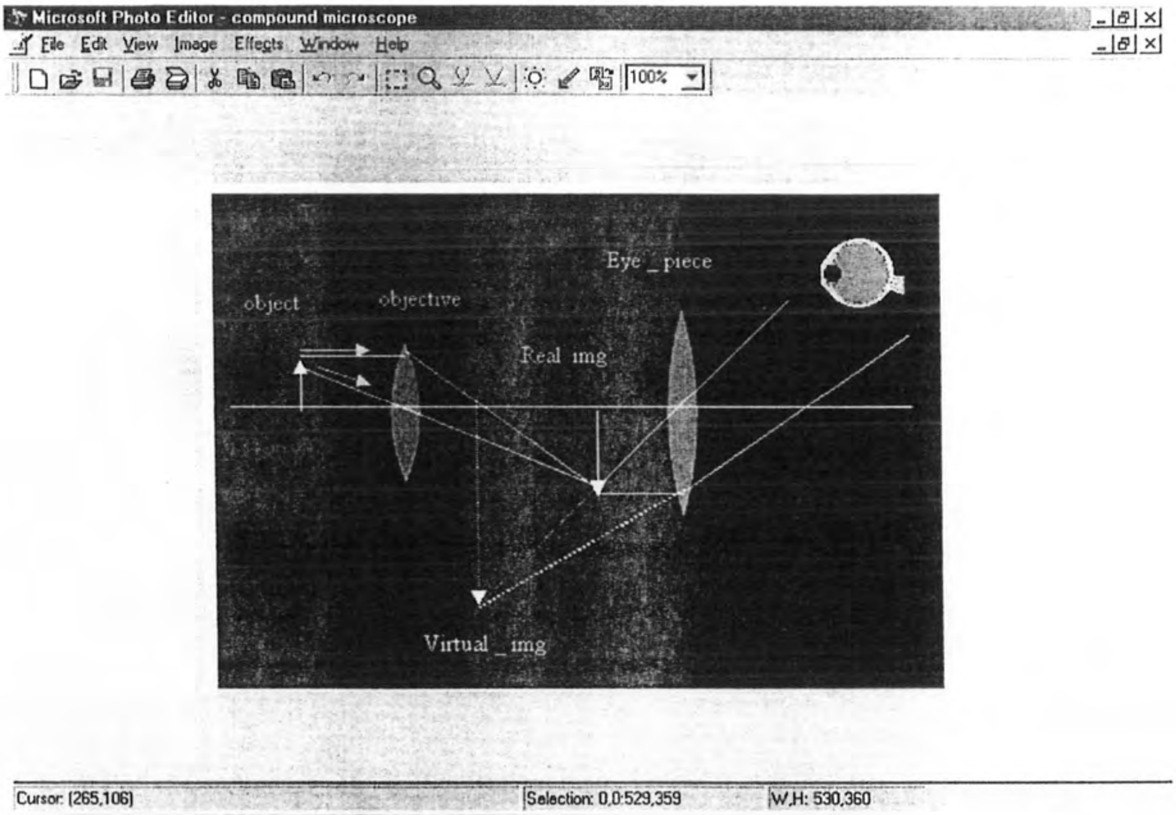
Calculate

length of pendulum= L = cm

Time for 20 vibrations


0
90
80
15X1mm

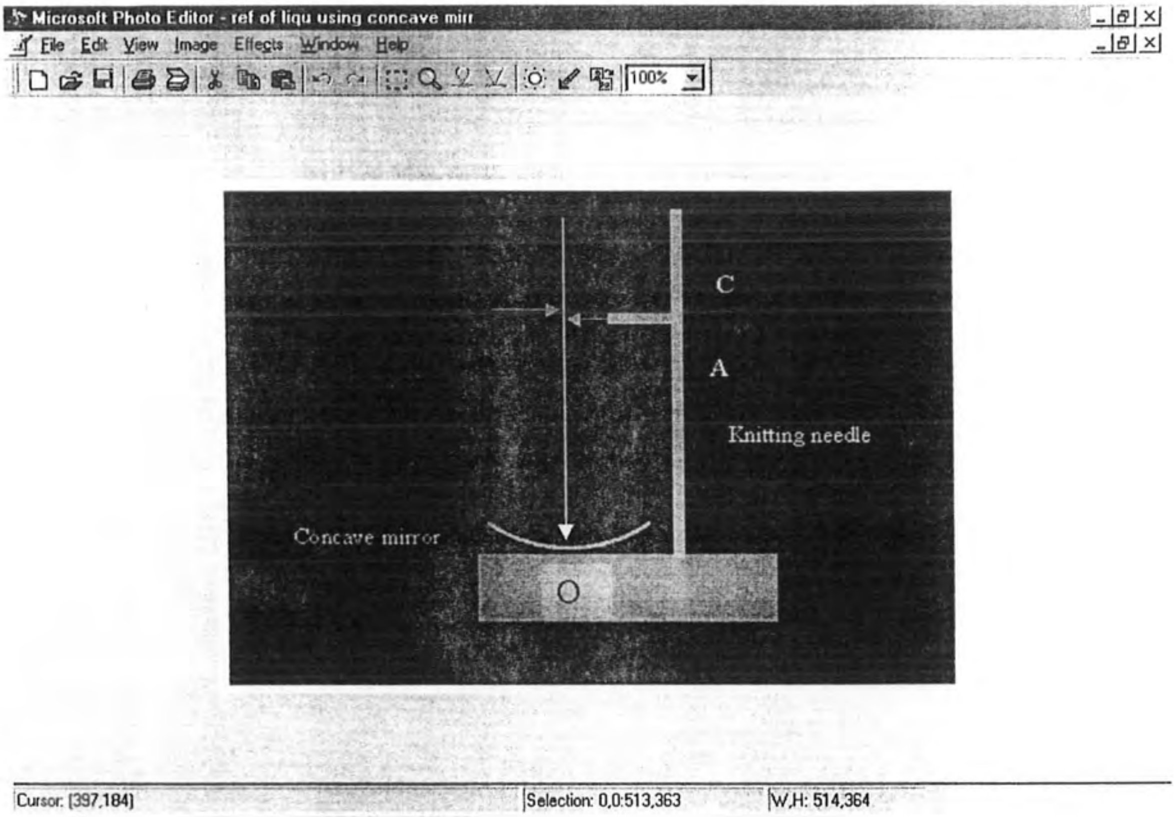

jaws for measuring inner dimensions
screw clamp
stem for measuring depths
main scale
vernier scale
jaws (for measuring outer dimensions)

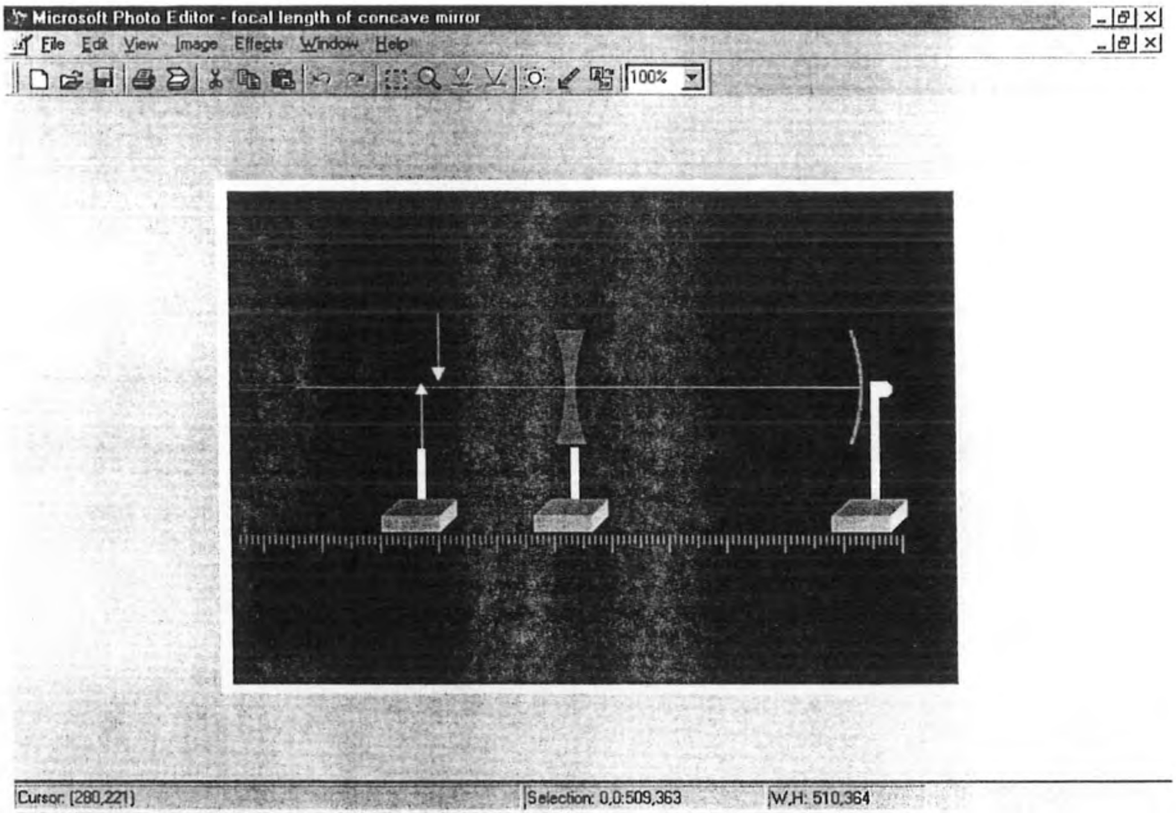

Microsoft Photo Editor - compound microscope
File Edit View Image Effects Window Help
100%
object
objective
Eye _ piece
Real img
Virtual _ img

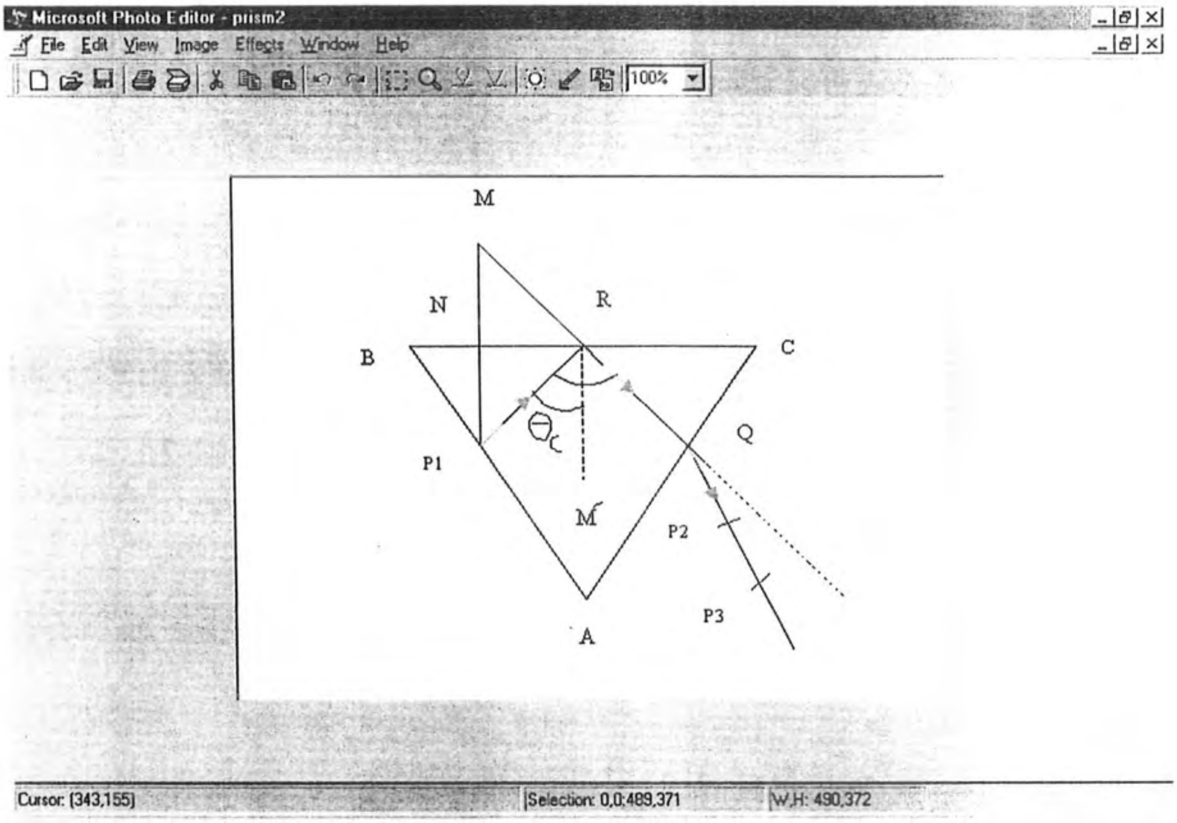
Cursor: (265,106)

Selection: 0,0:529,359

W,H: 530,360


Microsoft Photo Editor - ref of liqu using concave mirr
File Edit View Image Effects Window Help
100%
C
A
Knitting needle
Concave mirror
O
Cursor: [397,184]
Selection: 0,0:513,363
W,H: 514,364

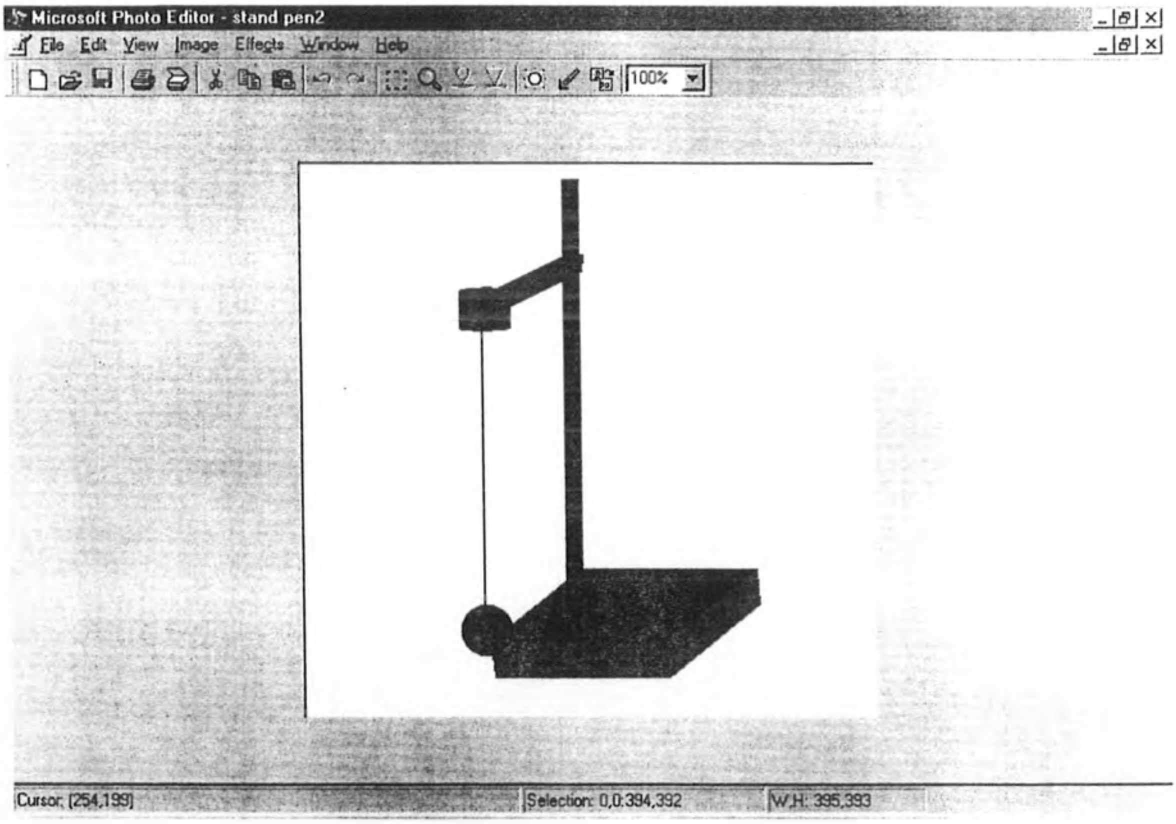

Microsoft Photo Editor - focal length of concave mirror
File
Edit
View
Image
Effects
Window
Help
100%
Cursor: (280,221)
Selection: 0,0:509,363
W,H: 510,364

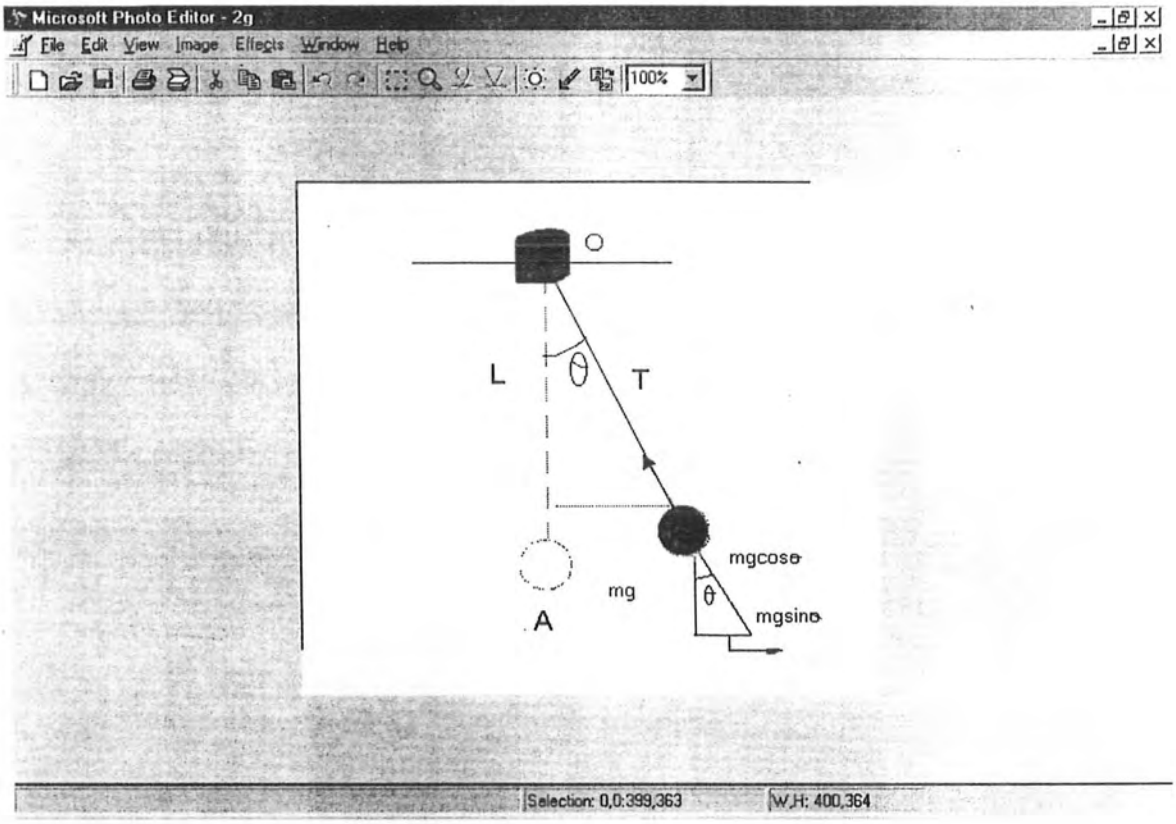

Microsoft Photo Editor - prism2
File Edit View Image Effects Window Help
100%
M
N
R
B
C
Q
P1
θ_c
Ḿ
P2
P3
A

Cursor: (343,155)

Selection: 0,0;489,371

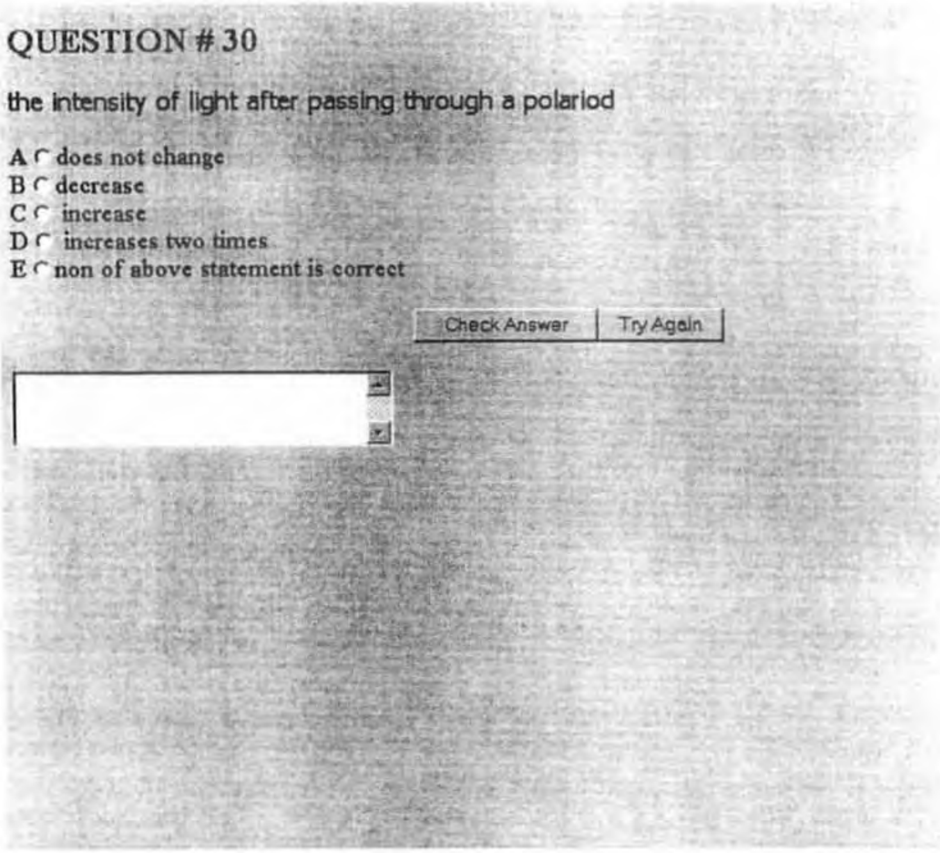
W,H: 490,372


Microsoft Photo Editor - stand pen2
File Edit View Image Effects Window Help
100%
Cursor: (254,199)
Selection: 0,0:394,392
W,H: 395,393


Microsoft Photo Editor - 2g
File Edit View Image Effects Window Help
100%
O
L
θ
T
mgcosθ
mg
θ
A
mgsinθ

Selection: 0,0:399,363

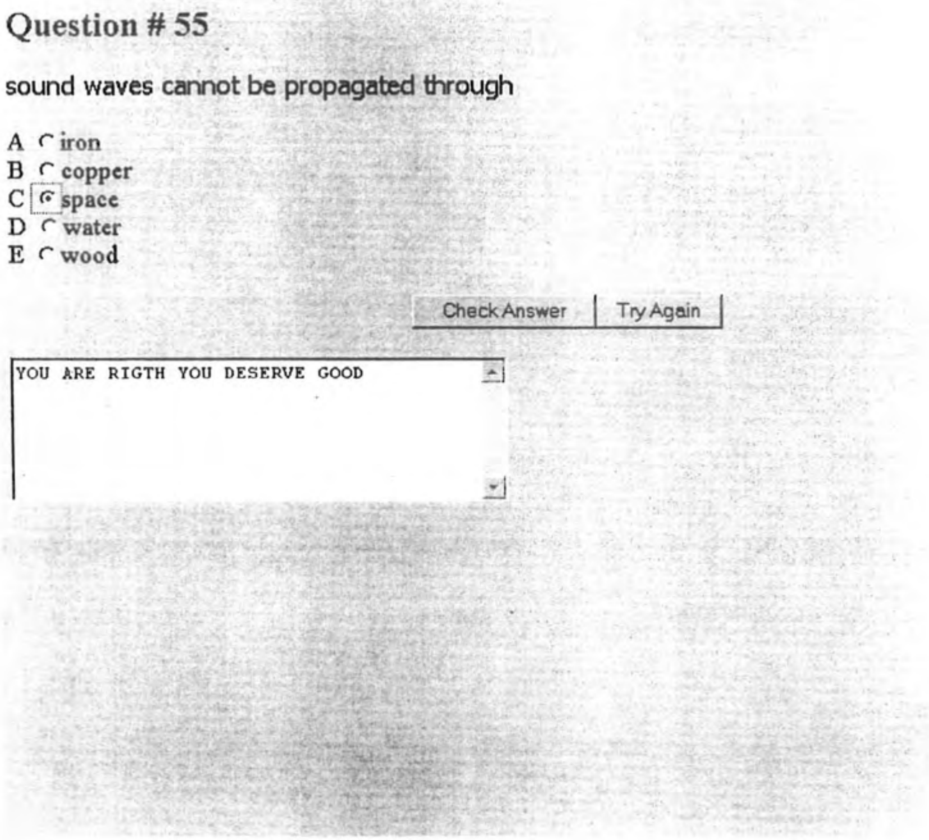
W,H: 400,364



QUESTION # 30

the intensity of light after passing through a polariod

A ○ does not change
B ○ decrease
C ○ increase
D ○ increases two times
E ○ non of above statement is correct

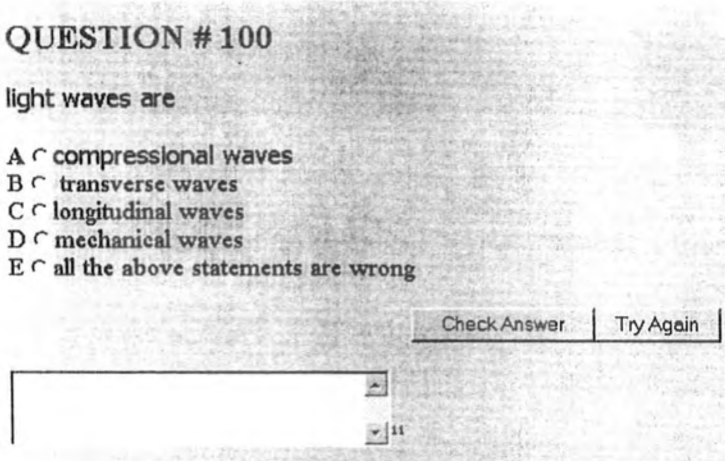


Question # 55

sound waves cannot be propagated through

A ○ iron
B ○ copper
C ● space
D ○ water
E ○ wood

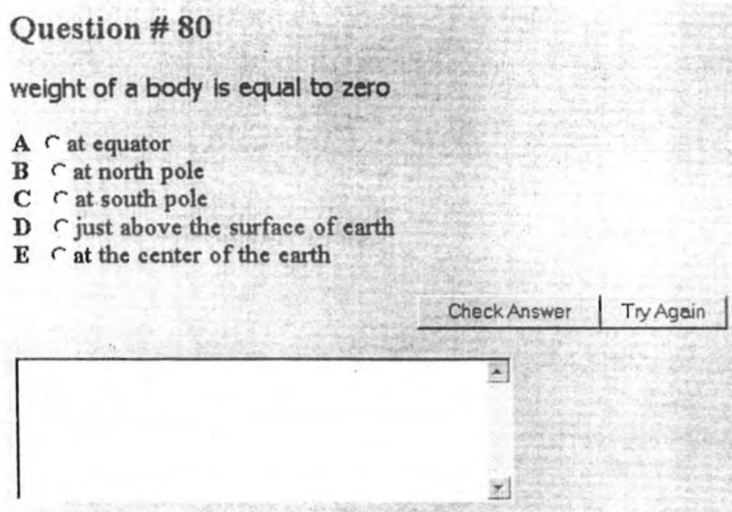
YOU ARE RIGTH YOU DESERVE GOOD



QUESTION # 100

light waves are

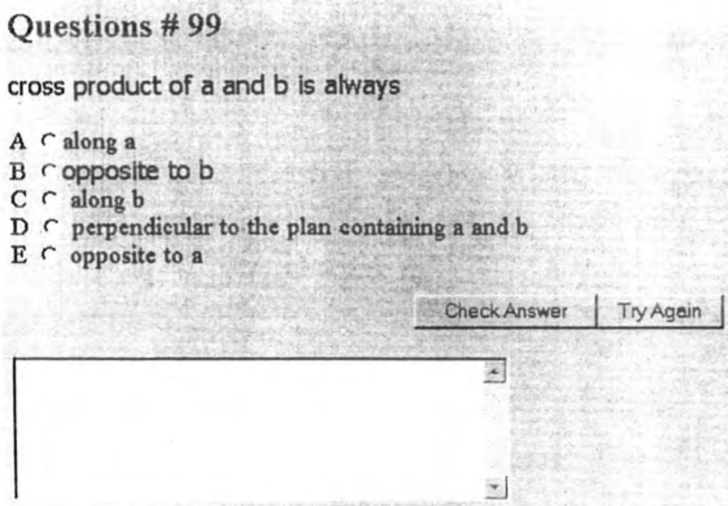
A ○ compressional waves
B ○ transverse waves
C ○ longitudinal waves
D ○ mechanical waves
E ○ all the above statements are wrong



Question # 80

weight of a body is equal to zero

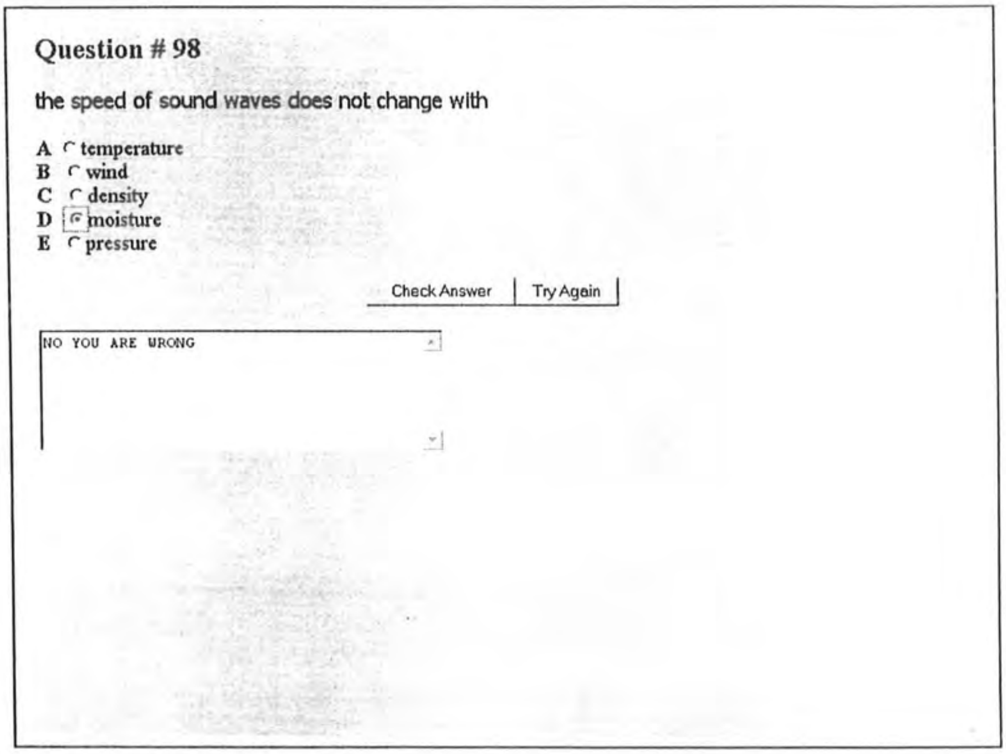
A ○ at equator
B ○ at north pole
C ○ at south pole
D ○ just above the surface of earth
E ○ at the center of the earth



Questions # 99

cross product of a and b is always

A ○ along a
B ○ opposite to b
C ○ along b
D ○ perpendicular to the plan containing a and b
E ○ opposite to a

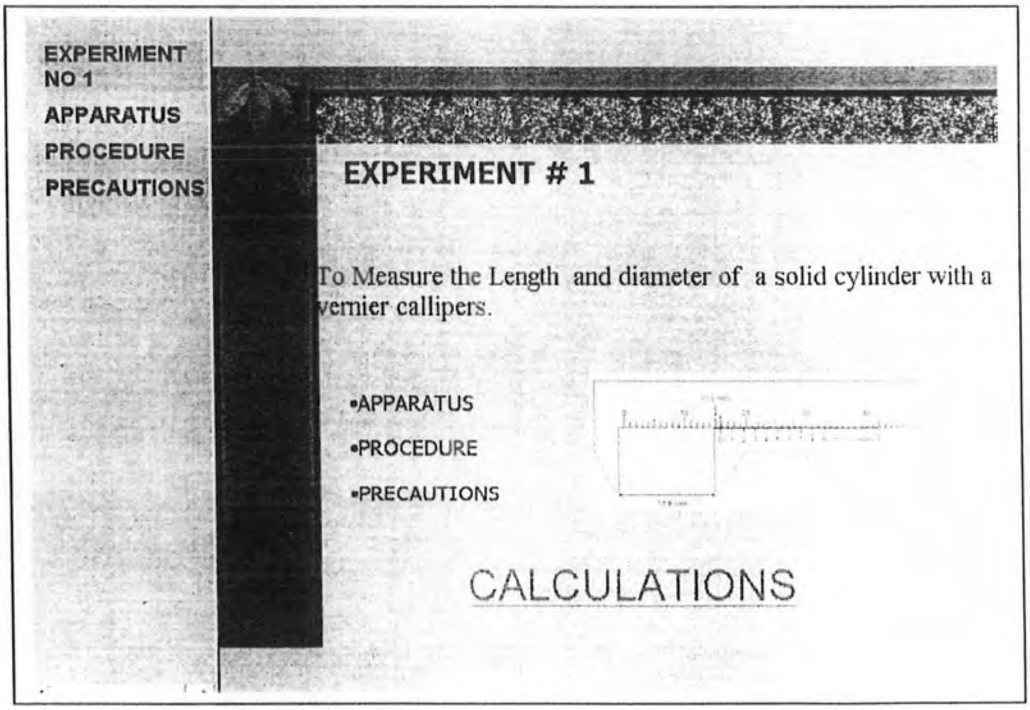


Question # 98

the speed of sound waves does not change with

A ○ temperature
B ○ wind
C ○ density
D ● moisture
E ○ pressure

```
NO YOU ARE WRONG
```

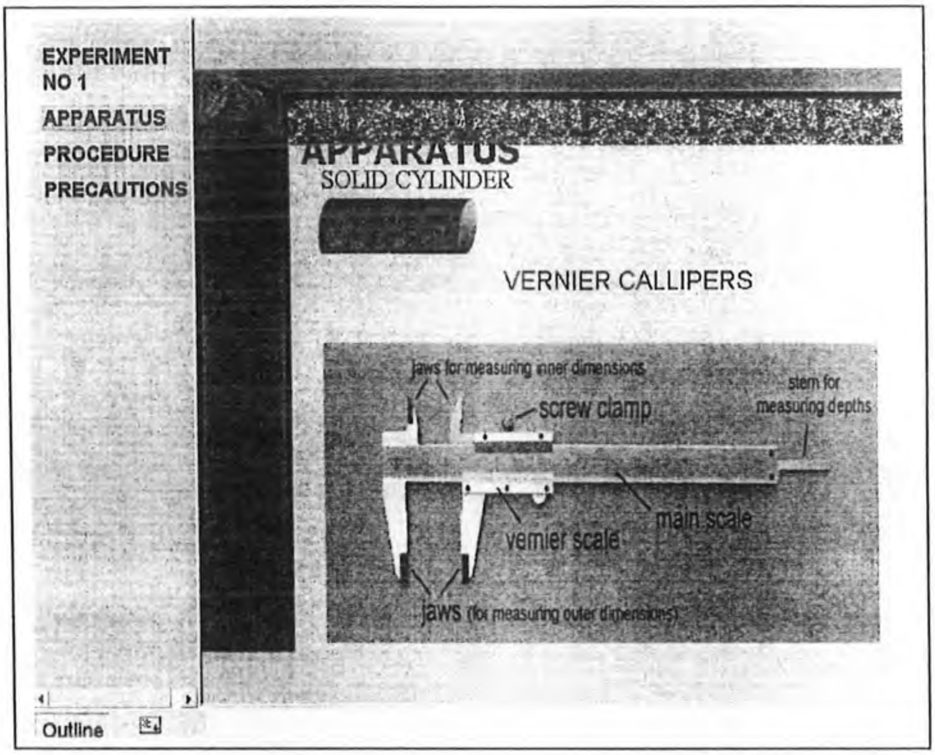


EXPERIMENT # 1

To Measure the Length and diameter of a solid cylinder with a vernier callipers.

- •APPARATUS
- •PROCEDURE
- •PRECAUTIONS

CALCULATIONS



EXPERIMENT NO 1

APPARATUS

PROCEDURE

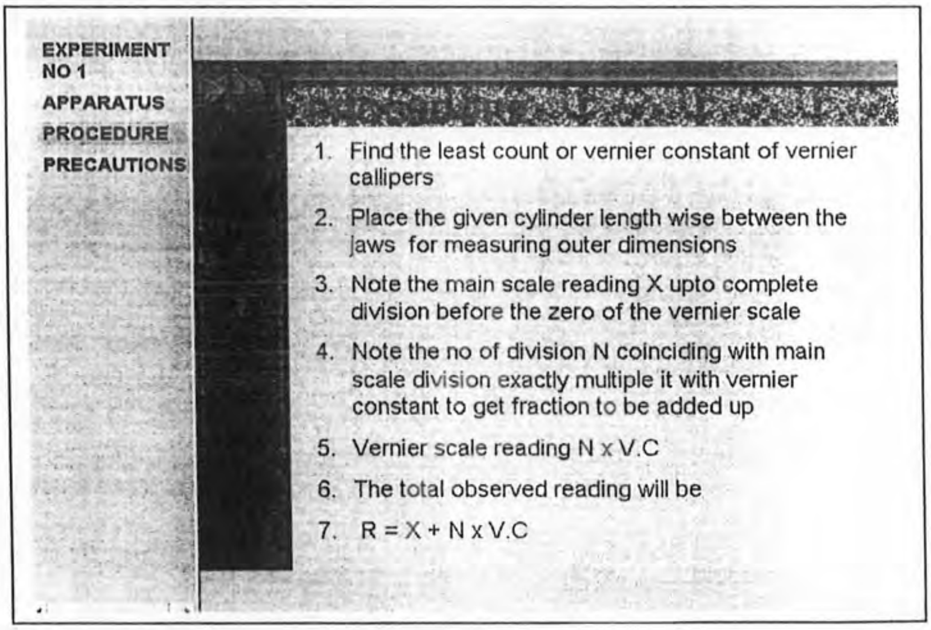
PRECAUTIONS

APPARATUS

SOLID CYLINDER

VERNIER CALLIPERS

jaws for measuring inner dimensions
screw clamp
stem for measuring depths
main scale
vernier scale
jaws (for measuring outer dimensions)
Outline



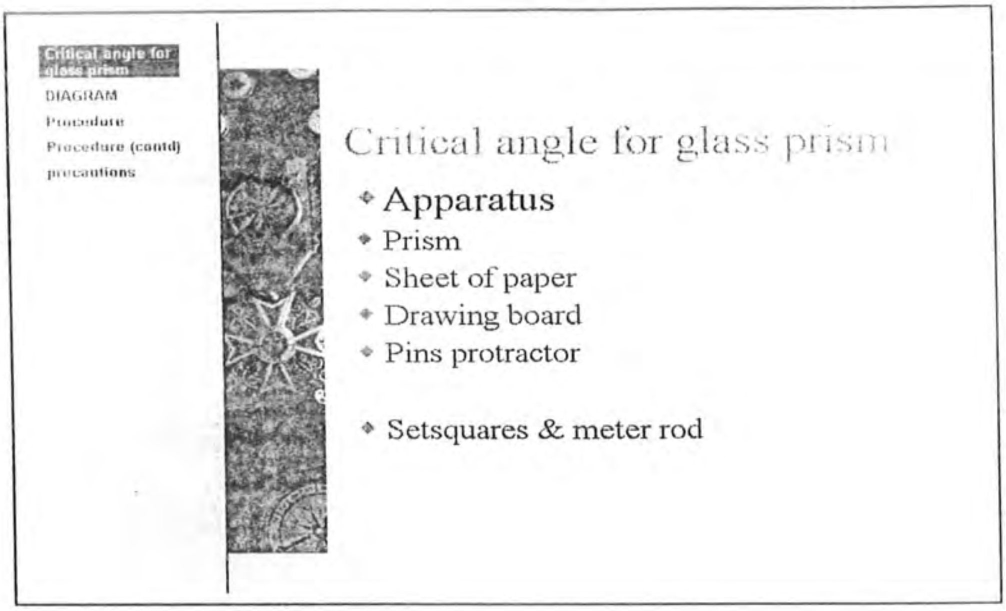
EXPERIMENT NO 1

APPARATUS

PROCEDURE

PRECAUTIONS

1. Find the least count or vernier constant of vernier callipers
2. Place the given cylinder length wise between the jaws for measuring outer dimensions
3. Note the main scale reading X upto complete division before the zero of the vernier scale
4. Note the no of division N coinciding with main scale division exactly multiple it with vernier constant to get fraction to be added up
5. Vernier scale reading N x V.C
6. The total observed reading will be
7. R = X + N x V.C



Critical angle for glass prism

DIAGRAM

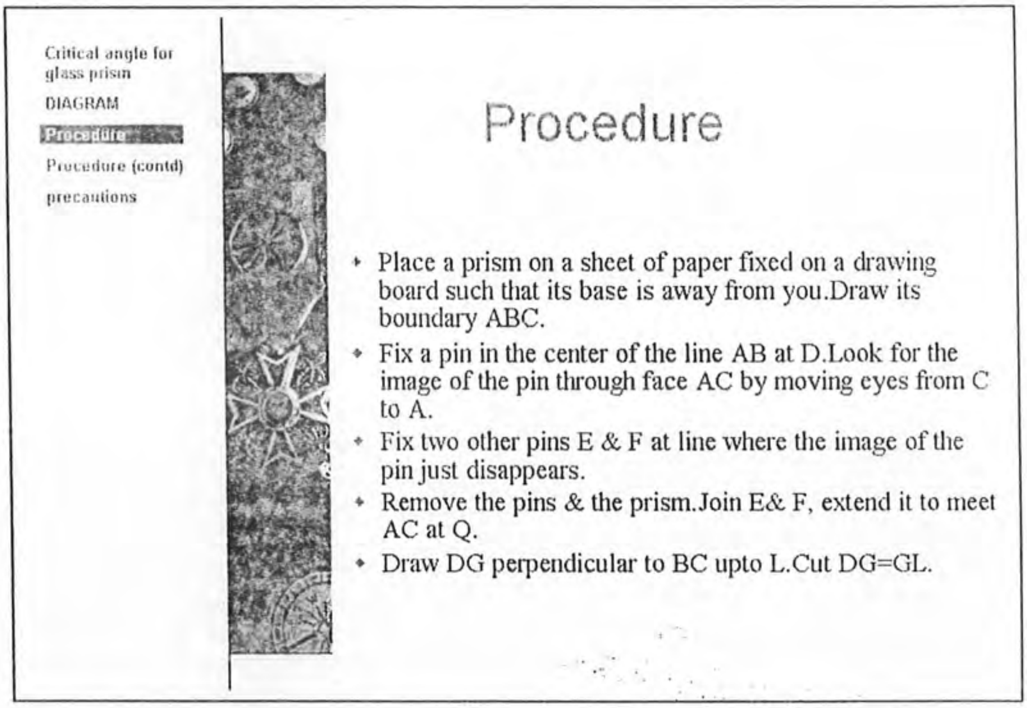
Procedure

Procedure (contd)

precautions

Critical angle for glass prism

- Apparatus
- Prism
- Sheet of paper
- Drawing board
- Pins protractor
- Setsquares & meter rod



Critical angle for glass prism

DIAGRAM

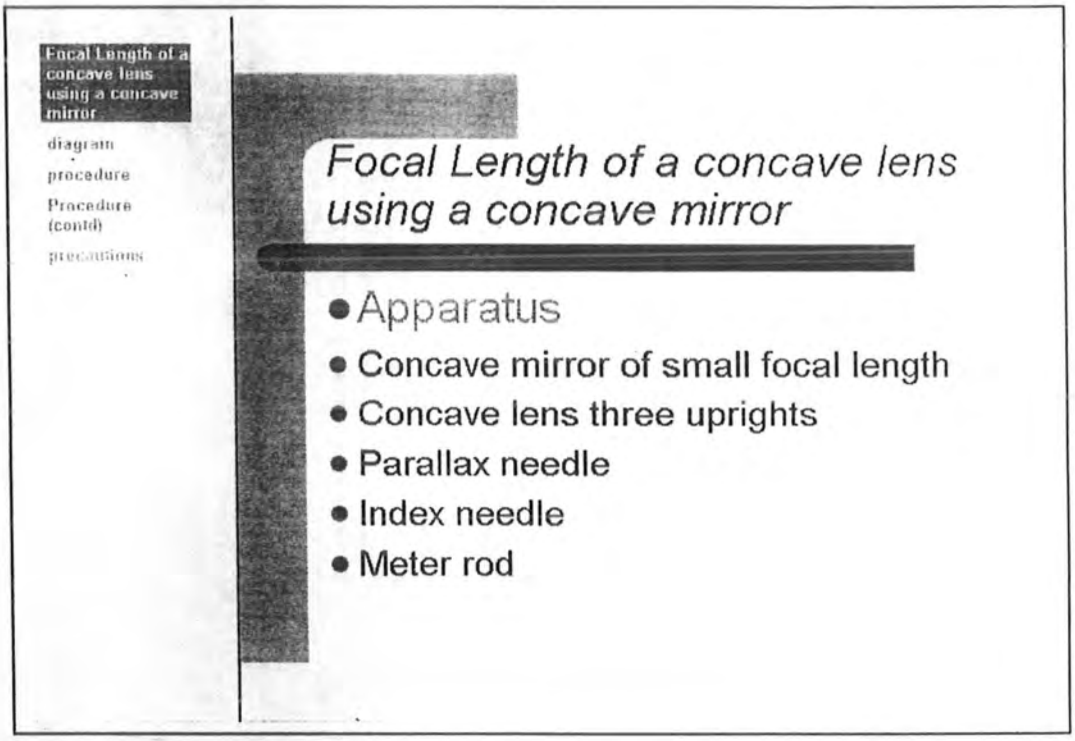
Procedure

Procedure (contd)

precautions

Procedure

- Place a prism on a sheet of paper fixed on a drawing board such that its base is away from you.Draw its boundary ABC.
- Fix a pin in the center of the line AB at D.Look for the image of the pin through face AC by moving eyes from C to A.
- Fix two other pins E & F at line where the image of the pin just disappears.
- Remove the pins & the prism.Join E& F, extend it to meet AC at Q.
- Draw DG perpendicular to BC upto L.Cut DG=GL.



Focal Length of a concave lens using a concave mirror

diagram

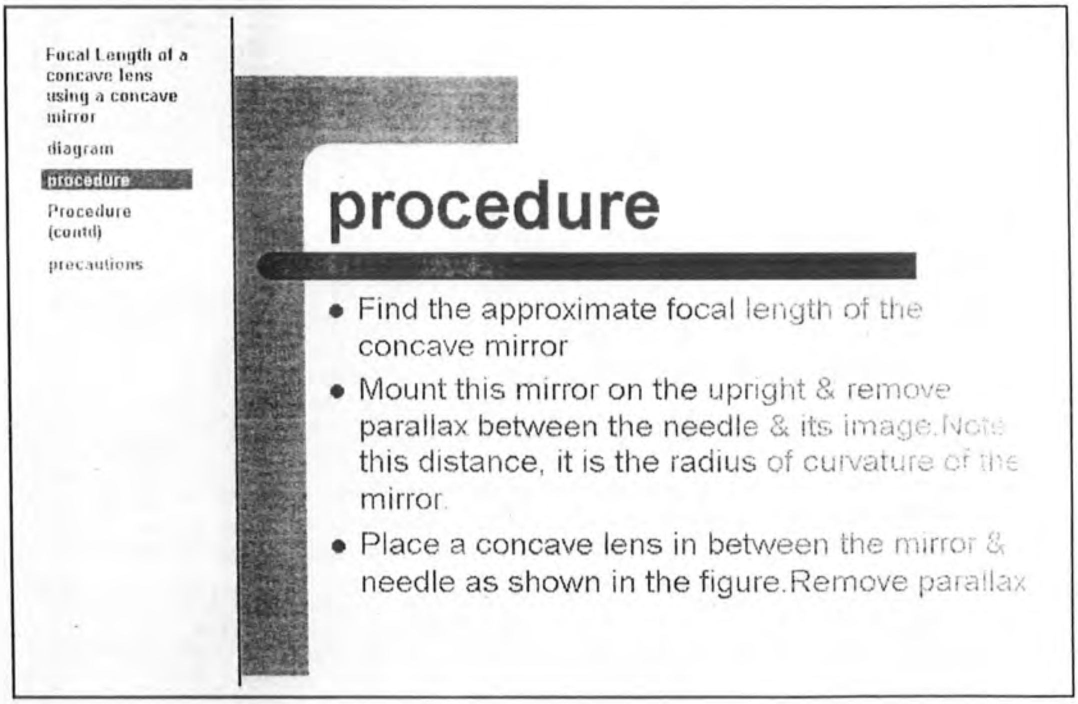
procedure

Procedure (contd)

precautions

Focal Length of a concave lens using a concave mirror

- Apparatus
- Concave mirror of small focal length
- Concave lens three uprights
- Parallax needle
- Index needle
- Meter rod



Focal Length of a concave lens using a concave mirror

diagram

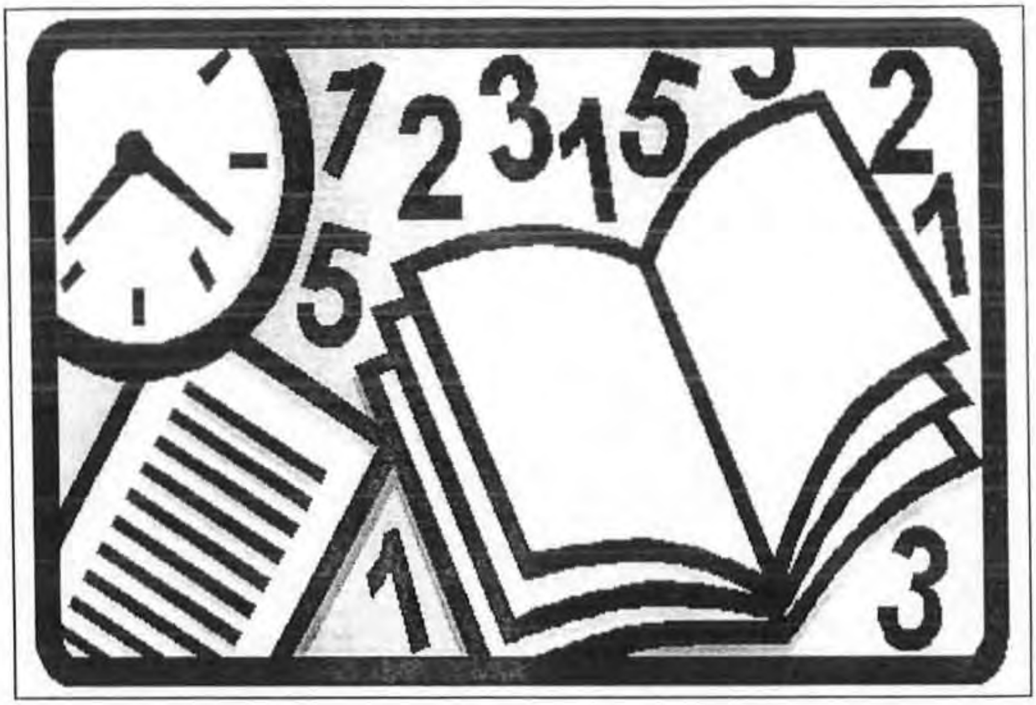
procedure

Procedure (contd)

precautions

procedure

- Find the approximate focal length of the concave mirror
- Mount this mirror on the upright & remove parallax between the needle & its image.Note this distance, it is the radius of curvature of the mirror.
- Place a concave lens in between the mirror & needle as shown in the figure.Remove parallax


7 2 3 1 5 5 2 1
5
1
3

Chapter 8

Conclusions

Objectives achieved

- The following objectives are so far achieved by this project.
- Use of computer technology in the classroom as well as in the laboratory.
- To facilitate the students of physics in carrying out their practical.
- To make the practical demonstrations interesting & more understandable.
- To decrease the time consumption in demonstrations.
- To facilitate the teacher by replacing his labor put into the apparatus setting & instructing.
- Helping the students to carry out the calculations on computer & getting the accurately calculated results readily in hands
- To provide a self-test in viva & related questions by an easy & interesting way.
- To inspire students for the use of computer technology in facilitating their study.

FUTURE EXPANTIONS

The scope of our work is ever expanding, because there could always be a better way of representing the practical demonstrations. Moreover the syllabus keeps on revising & changing day by day. we therefore believe in a constant updating process.

As with the passing time the computer technology is expanding & improving, the current work would need to be updated & improved the span of work can also be expanded to include the practicals of other levels such as matric & BSc etc.
Secondly the animations included in this work can be made more & more detailed & advanced. Since this part of the work is more related to creativity & innovation, it shall always have a room for improvements & would require to be made better & better.
And we can choose any other language, which is window, based for this project.
It is actually very creative work to make all things related to this project.

Recommendations

This project presents an easy to follow laboratory-based demonstration at the college level. It is therefore required that the students trying to benefit from it should have a considerable understanding & appreciation of the basic physics of HSSC level. In order to enhance the applicability and effectiveness of this program we recommend that the students must have an adequate knowledge of the basic physics principals. The point of this recommendation is that a student benefiting from this program needs to acquire:

1. A basic understanding of physics principles and appreciation of scientific methodology & scientific literacy.
2. The ability to analyze, measure & record scientific observations.
3. The student should comply with all laboratory rules and precautions.
4. The student should practice the practical himself again and again in order to acquire expertise.