Savitribai Phule Pune University

M. C. A. Part II

(Under Science Faculty)

CA -405

SEMESTER IV

Name _____

College Name _____

Roll No. _____ Division _____

Academic Year _____

Prepared By

Dr. Shailaja C. Shirwaikar

Head, Dept of Comp Sc, Nowrosjee Wadia College

Prof. Aparna Gohad

Assistant Professor, Nowrosjee Wadia College

Preface

This Lab Book Contains the assignments to be carried out related to Event Driven programming(SDK), Computer Graphics and Advanced Java as part of CA 405 Lab course and also guidelines for mini projects related to SDK and Graphics. The intention is to bring uniformity in conducting the lab sessions across various affiliated colleges. The assignments are designed so that the theory concepts in the syllabus are broadly covered. Mini project guidelines and some mini project ideas are outlined however the students need not limit their choice to these ideas alone. They are specified in order to demonstrate the complexity level of expected mini project. Users can always bring to our notice any additions deletions which can be subsequently incorporated into the Lab Book. Book is always going to remain in digital form and available on the Department of Computer Science, University of Pune, website. We are all indebted to Dr. Vilas Kharat, Chairman, Board of studies in Computer Science for continuous encouragement, support and guidance.

Dr. Shailaja C. Shirwaikar

Member, BOS Computer Science

Table of contents

Introduction

1. About the work book

This workbook is intended to be used by M. C. A. Part II (Under Science faculty) students for the CA 405 lab course. The course is indented to cover assignments related to three theory subjects namely CA 401 : computer Graphics, CA 402 Windows SDK and CA: 403 Advanced Java.

The objectives of this book are

1) Defining clearly the scope of the course

2) Bringing uniformity in the way the course is conducted across different colleges

3) Providing ready reference for students while working in the lab so that they are able to carry out the assignments without the help of instructors

4) Catering to the need of slow paced as well as fast paced learners by defining basic set of assignments to be carried out by slow paced learners and by providing challenging problems to fast paced learners

1.1. Instructions to the Lab administrator

For computer graphics OpenGL should be installed on Linux server so that students can write programs in C using OpenGL. For Advanced Java, Tomcat apache server need to be installed. Visual Studio need to be installed for carrying out SDK assignments.The required Packages and installation procedure is given below

a) OpenGL

1.Required packages

glut-3.7-8.i386.rpm

glut-devel-3.7-8.i386.rpm

2.Installation Procedure:

Run following command on ***terminal*** to install package

rpm –ivh glut-3.7-8.i386.rpm

rpm –ivh glut-devel-3.7-8.i386.rpm

b) Apache Tomcat Server

1.Required Packages

jdk-7u65-linux-i586.gz

apache-tomcat-8.0.15.tar

2.Installation Procedure:

- Extract above packages in ***/opt*** directory
- Set path for java and Tomcat in ***.bash_profile*** file (this file is located in **/root**)

e.g.

PATH=$PATH:/opt/jdk1.7.0_65/bin
export PATH

CLASSPATH=$CLASSPATH:/opt/tomcat/lib/servlet-api.jar
export CLASSPATH

JAVA_HOME=/opt/jdk1.7.0_65/
export JAVA_HOME

cd /opt/tomcat/bin
sh startup.sh

c) For SDK

1. Install Microsoft Visual Studio 2008/2010

Assignment Completion Sheet

Lab Course CA 405			
Section I – Computer Graphics			
Sr. No	**Assignment Name**	**Marks (out of 5)**	**Signature**
1	Assignment I		
2	Assignment II		
3	Assignment III		
4	Assignment IV		
Total (out of 20)			
Total (out of 10)			
Section II – Windows (SDK)			
5	Assignment I		
6	Assignment II		
7	Assignment III		
8	Assignment IV		
Total (out of 20)			
Total (out of 10)			
Section III – Advanced Java			
9	Assignment I		
10	Assignment II		
11	Assignment III		
12	Assignment IV		
13	Assignment V		
14	Assignment VI		
Total (out of 30)			
Total (Out of 10)			

Section I – Computer Graphics

Introduction to OpenGL

The hardware dependency of graphics is handled by using operating system as an interaction interface. Programmers use an Application Programming Interface(API). OpenGL is a software interface (set of APIS) to graphics hardware, It provides a set of functions that allow graphics programmers to specify operations needed to produce complex 2D and 3D graphics.

OpenGL uses Event Driven programming. Program responds to various events such as mouse clicks or keyboard button presses. The system maintains an event queue for a window. The messages are added to the queue when event occurs. The program deals with these events by executing a set of callback functions that can handle the events. Procedural programs have the structure ' do this and then do this' while event driven programs have the structure ' do nothing till the event occurs and then do what is specified for that event'. The system patiently waits in a loop for an event to occur. OpenGL provides GLUT library that handles event management, opening windows, managing menus etc.. Programmer has to register the call back function of his choice which should contain the code specific to events of interest.

Writing Graphics programs using OpenGL

Every graphics program performs some initializations such as choosing display mode and setting the co-ordinate system. In window based system a screen window is used to display pictures and the co-ordinate system is attached to the window, openGL handles the changes when window size is changed and is not programmer's responsibility. Program uses a set of primitive operations for drawing i.e adding points, lines or polygons to the picture.

The graphics program structure is given below. It has three main parts. An initialization function which performs all the initialization. A display function that contains the code required to display picture according to the requirements of the application. The main function that contains the necessary code for creating and displaying window, registering the display function as a callback function and also a call to the initialization function.

```
// include the OpenGL libraries
void myinit(void)
{
// do all the necessary initializations
}
void myDisplay(void)
{
// this is the callback function that will contain the application specific code
}
void main(int argc , char ** argv)
{
// will contain the standard code using GLUT library functions
}
```

OpenGL library comprises of four components

- Basic GL – provides basic set of functions
- GLUT – GL utility toolkit- provides functions for opening windows , managing menus and managing events
- GLU – GL utility library – provides high level functions for advanced graphics
- GLUL – user interface library – provides support for sophisticated menus

A simple graphics program for creating and presenting a picture will include the first three libraries. The following include statements are essential.

```
#include <GL/gl.h>
#include <GL/glu.h>
#include <GL/glut.h>
```

Basic GL functions

Basic GL library functions start with gl. Most gl functions have several versions depending on the number and type of arguments

glColor3f(..) indicates no of arguments as 3 and data type as float

glVertex2iv(..) indicates no of arguments as 2 , data type as integer and that the arguments will be provided not individually but in vector form. All commands do not have vector form

The table shows the basic openGL data types and the corresponding C data types.

Character	ubC-Language Type	OpenGL Type
b	signed char	GLbyte
s	Short	GLshort
i	Int	GLint , GLsizei
f	Float	GLfloat, GLclampf
d	Double	GLdouble, GLclampd
ub	unsigned char	GLubyte, GLboolean
us	unsigned short	GLushort
ui	unsigned int	GLuint,GLenum
	Void	GLvoid

Basic Initialization Code is given below

```
void myinit(void)
{
glClearColor(1.0,1.0,1.0,0.0); //set the background color to bright white ,
// last parameter specifies transparency  0.0 indicates  no transparency
glColor3f(0.0, 0.0, 0.0);//set the drawing color to black;
glPointSize(4.0);//set the point size to 4 by 4 pixels
glMatrixMode(GL_PROJECTION); //set up the co-ordinate system
glLoadIdentity();
gluOrtho2D(0.0, 640.0, 0.0, 480.0);
}
```

The function glClearColor is a framebuffer function used to set the initial or background color. All pixels of the frame buffer are set to the same color

It has four parameter all of the type Glclampf, the values are of type float and clamped to the range [0,1]. The first three parameters specify the red, green and blue component and the last parmeter specifies alpha value which indicates the transparency

glClearColor(1.0,1.0,1.0,0.0);

sets the background color to bright white, 0.0 indicates no transparency

The function glClear() uses a single argument and is used for resetting buffers to preset values.

glClear(GL_COLOR_BUFFER_BIT)

glColor has 3 or 4 parameters and data type can be of any data type.

The three parameters specify red green and blue component and the fourth parameter specifies the intensity. If the last parameter is not specified , the intensity value is set to 1.0 that is full intensity. It is used for setting the current color and can be called any where to change the current color.

glColor3f(1.0, 0.0,0.0); // sets the color to red

glColor3f(0.0, 1.0,0.0); // sets the color to green

glColor3f(0.0, 0.0,1.0); // sets the color to blue

glColor3f(0.0, 0.0,0.0); // sets the color to black

glColor3f(0.7, 0.7,0.7); // sets the color to bright gray

glColor3f(0.2, 0.2,0.2); // sets the color to medium gray

glColor3f(1.0, 1.0,1.0); // sets the color to white

glColor3f(1.0, 0.0,1.0); // sets the color to magenta

glColor3f(0.0, 1.0,1.0); // sets the color to cyan

The callback function given below contains the main application specific code. Here it is putting a single pixel on the screen.

```
void myDisplay(void)
{
glClear(GL_COLOR_BUFFER_BIT); //clear the entire window to background color
glBegin(GL_POINTS); //specifies a set of points
glVertex2i(40,40); // plot the point with two integer arguments
glEnd();
glFlush(); //the entire data is to be processed and sent for display
}
```

glBegin and glEnd are used to delimit the vertices of a primitive operation. The single argument to glBegin specifies the mode in which the vertices are to be interpreted.

GL_POINTS treats each vertex as a single point
GL_LINES treats each pair of vertices as an independent line segment
GL_LINE_STRIP draws a connected group of line segments
GL_LINE_LOOP draws a connected group from first to last
The other possibilities are - GL_TRIANGLES, GL_TRIANGLE_SRIP, GL_TRIANGLE_FAN, GL_QUADS, GL_QUAD_STRIP, GL_POLYGON
glEnd() has no arguments

```
void main(int argc, char **argv)
{
glutInit(&argc, argv);// initializes the grahics toolkit
glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB); //single display buffer and RGB mode
glutInitWindowSize(640,480);// initial window size
glutInitWindowPosition(350,350);//initial position of windows left corner
glutCreateWindow("My First Attempt");//opens a window with title
glutDisplayFunc(myDisplay);// Registering the display function
myinit();// call the initialization function for performing necessary initializations
glutMainLoop();//go into a perpetual loop
}
```

.

GLUT – the GL utility toolkit provides all the functions necessary for Event-driven programming. The function glutinit initializes the toolkit and it uses command line arguments.The glutInitDisplayMode function specifies the display mode by oring some built-in constants. The glutInitWindowSize function is then used to specify the window size and gluInitWindowPosition specifies where the window should be positioned on the screen.

The glutCreateWindow function is used to create and display the window after giving a call to the registered callback function which is registered by the function glutDisplayFunc. After calling the initialization function , the program goes into a perpetual loop waiting for an event to occur.

Complete Simple graphics program is given below

```
#include <math.h>
#include <stdio.h>
#include <GL/gl.h>
#include <GL/glu.h>
#include <GL/glut.h>
GLint x,y;
void myinit(void)
{
glClearColor(1.0,1.0,1.0,0.0);
glColor3f(0.0, 0.0f, 0.0f);
glPointSize(4.0);
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluOrtho2D(0.0, 640.0, 0.0, 480.0);
}
```

```
void myDisplay(void)
{int i;
glClear(GL_COLOR_BUFFER_BIT);
glBegin(GL_POINTS);
glVertex2i(40,40);
glEnd();
glFlush();
}
void main(int argc, char **argv)
{
glutInit(&argc, argv);
glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
glutInitWindowSize(640,480);
glutInitWindowPosition(350,350);
glutCreateWindow("My First Attempt")e
glutDisplayFunc(myDisplay);
myinit();
glutMainLoop();
}
```

The compilation and execution procedure is as follows

```
$cc first.c -lGLU -lglut -lGL -o first
$./first
```

For every program, initialization function and main function remain almost the same but the display function will vary.

The display functions for some simple programs are given below

Program1: Display 50 random points

```
void myDisplay(void)
{int i;
GLint x,y;
```

```
glClear(GL_COLOR_BUFFER_BIT);//clear the entire window to background color
glBegin(GL_POINTS);//specifies a set of points
for(i=0; i<50; i++)
{
x=rand()%630;
y=rand()%470;
glVertex2i(x,y);// plot the point with two integer arguments
}
glEnd();
glFlush();//the entire data is to be processed and sent for display
}
```

Program 2 : Divide the display surface into four quadrants by drawing x and y axis

```
void myDisplay(void)
{
int i;
glClear(GL_COLOR_BUFFER_BIT);//clear the entire window to background color
glBegin(GL_LINES);//specifies a set of lines
glVertex2i(0,240);// starting point of the line
glVertex2i(640,240);// end point of the line
glVertex2i(320,0);
glVertex2i(320,480);
glEnd();
glFlush();//the entire data is to be processed and sent for display
}
```

Program 3: Drawing a simple house

```
void myDisplay(void)
{int i;
glClear(GL_COLOR_BUFFER_BIT)r
glBegin(GL_LINE_LOOP);//specifies a set of lines connected lines
glVertex2i(40,40);
glVertex2i(40,240);
glVertex2i(220,420);
glVertex2i(420,240);
glVertex2i(420,40);
glEnd();
glBegin(GL_LINE_STRIP);//specifies a open set of lines
glVertex2i(280,40);
glVertex2i(280,200);
glVertex2i(200,200);
glVertex2i(200, 40);
glEnd();
glFlush();
```

Assignment 1: Simple display of graphics using point , line, polygon primitives

1) Create the following graphics using simple graphics primitives (any FIVE)

a) Display random 50 points (4x4 pixel size) with random colors (use rand() function and glBegin(GL_POINTS)) [1 marks]

b) Divide the screen area with four quadrants and divide each axis into intervals of size 10 (use glBegin(GL_LINES) [1 marks]

c) Display a grid along with axis that divides the entire area into 50 by 50 size boxes. The grid should be in light green color[1 Marks]

d) Display a picture containing a two storeyed house(use glBegin(GL_LINE_LOOP) and glBegin(GL_LINE_STRIP)[1 Marks]

e) Display a chessboard (use gRecti(..))[1 marks]

f) Display sine/cosine function [1 marks]

Each correct program carries 1 mark, Maximum marks for the assignment is 5

Assignment Evaluation

No of Completed programs : Marks :

Signature of the Instructor

++

Assignment 2: Display of graphics by invoking user written functions

Write a program to draw a scene. Program should use graphics primitives to draw objects. Program should have routines to create complex objects using graphics primitives. These routines should be invoked to create the final scene.

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

Signature of the Instructor Date

++

Assignment 3: Implementing primitive algorithms (line , circle etc.)

Implement any one of the following primitive algorithms

a) Implement Bresenham line drawing algorithm

b) Implement Bresenham circle drawing algorithm

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

Signature of the Instructor Date

++

Assignment 4: Implementing different transformations

Implement (any THREE) basic transformations as functions and apply these transformations on basic objects and display the transformed objects

a)Translation

b) Rotation about the origin/pivot point

c) Scaling about the origin/ fixed point

d) Reflection about x-axis/ y-axis/ line y=x

e) Sheer along x-axis/y-axis

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

Signature of the Instructor Date

++

Graphics mini-Project Guidelines

The project is to be carried out with a team size of Two or Three. The graphics mini project should solve a small problem completely or a larger problem partially. There should be use of display graphics using Open GL primitives.

Some of the Project ideas are given below.

I) Write a program to read data from a file and displays the picture. Generate few such interesting files. You can design your own file format. For example if your picture contains only polylines then the only entry on first line of the file will specify number of polylines. This will be followed by line entries for each polyline. The first line entry will indicate number of points in the polyline followed by co-ordinates of each point on the polyline in successive line entries. You can generalize the file format to include the other primitive types, colour values etc. (Refer case study 2.4 in [1])

II) Write a program that generates a Maze of R rows and C columns and find and display the path from start to end (Refer case study 2.5 in [1])

III) Write a program that creates tiles that is a square window with some interesting geometrical pattern /motif onto it. Produce tiling of the display screen by placing tiles all over the screen (Refer case study 3.6 in [1])

IV) Write functions to draw scatter plot, bar chart, line chart and pie chart for a given set of data points as input or in a file. Write a program that can demonstrate these functions

V) Write functions to draw some basic objects. Write scaling transformations so that picture can be enlarged to large size or compressed to smaller size. Accept an integer argument, the object type, the size(low, medium, large) and colour codes (RGB) as input and display n objects of the given size and colour in a line.

References:

[1] Computer Graphics using OpenGL , 3rd Edition By F. S. Hill , Stephen M. Kelly

Introduction to windows (SDK)

Windows software development kit (SDK) helps software developers focus on writing applications by providing an extensive set of system defined functions that provide access to Operating System features, collectively called as API(Application Programming Interface.

Header Files

API contains several hundred functions with easy to remember meaningful names, that application program can call. All these functions are declared in header files. The main header file is windows.h, which in turn contains other header files.

Applications should have a GUI consistent with other applications and operating system. A subset of API called GDI (Graphics Device Interface) provides device independent graphics support.

Windows.h is master header file which includes other header files , some of which include other header files. Important header files are

WINDEF.H Basic type definitions
WINNT.H Type definitions for unicode support
WINBASE.H kernel functions
WINUSER.H user interface functions
WINGDI.H Graphics Device Interface Functions

The Main Function

Every Win32-based application must have an entry-point function. The name commonly given to the entry point is the WinMain function. WinMain is the entry point of any windows program.

In a Windows program, the WinMain function completes the following steps:

- Registering the window class
- Creating the main window
- Entering the message loop

In Windows, a window is a rectangular area on the screen that receives input and displays output in the form of text and graphics. MessageBox is a special-purpose window but with limited flexibility. It is a type of dialog box. There are application windows, dialog boxes, child windows or control windows like buttons, check boxes, list boxes etc. User interacts with these windows using keyboard or mouse.

Window Procedure

Every window object has an associated window procedure which is a function either defined in the program or in some dynamic link library. Windows(OS) sends a message to a window by giving a call to its window procedure. Every window must have a window

procedure. The name of the window procedure is user-defined.

LRESULT WINAPI MainWndProc(HWND, UINT, WPARAM, LPARAM);

The window procedure receives messages from Windows. These may be input messages or window-management messages from Windows. You can optionally handle a message in your window procedure or pass the message to Windows for default processing by calling the DefWindowProc function. For example the application can process the WM_PAINT, and WM_DESTROY messages, using a switch statement that is structured as follows:

```
switch( msg ) {
case WM_PAINT:
...
case WM_DESTORY:
...
default:
        return( DefWindowProc( hWnd, msg, wParam, lParam ));
```

The entry point function almost remains the same for different applications while window procedure varies from application to application

A complete Windows(SDK) program is given below

```
/********************* Header Files   *********************/
#include <windows.h>
#include <tchar.h>
/********************* String declarations   *********************/
static TCHAR szClassName[]=_T("Mywindowclass");
static TCHAR szTitle[] = _T("Hello");
/********************* Prototypes   ***********************/
LRESULT WINAPI MainWndProc( HWND, UINT, WPARAM, LPARAM );
/************************   Win Main program  ***********************************/
int PASCAL WinMain( HINSTANCE hInstance,      HINSTANCE hPrevInstance,
    LPSTR lpszCmdLine,      int nCmdShow )
{
  WNDCLASS wc;
  MSG msg;
  HWND hWnd;
************************  Filing  the Window class structure  ***********************************/
  if( !hPrevInstance )
  { wc.lpszClassName = "HelloWin";
wc.lpfnWndProc = MainWndProc;
wc.style = CS_OWNDC | CS_VREDRAW | CS_HREDRAW;
wc.hInstance = hInstance;
wc.hIcon = LoadIcon( NULL, IDI_APPLICATION );
wc.hCursor = LoadCursor( NULL, IDC_ARROW );
wc.hbrBackground = (HBRUSH)( COLOR_WINDOW+1 );
wc.lpszMenuName = NULL;
wc.cbClsExtra = 0;
wc.cbWndExtra = 0;
}
/************************  Registering the window class ***********************************/
if(!RegisterClass( &wc ))
```

```
{ MessageBox(NULL,TEXT("Window Registration Failed"), TEXT("Error!"),
MB_ICONEXCLAMATION|MB_OK);
 return 0; }
**********  Creating the WindowWin Main program  **********/
hWnd = CreateWindow( "HelloWinClass",
   "Hello Program",
   WS_OVERLAPPEDWINDOW|WS_HSCROLL|WS_VSCROLL,    0,    0,
   CW_USEDEFAULT,    CW_USEDEFAULT,    NULL,    NULL,    hInstance,   NULL
);
if(hWnd == NULL)
{ MessageBox(NULL, TEXT("Window Creation Failed"), TEXT("Error!"),
MB_ICONEXCLAMATION|MB_OK);
 return 0; }
**********  Showing the window  **********/
ShowWindow( hWnd, nCmdShow );
UpdateWindow(hWnd);
**********  Message loop  **********/
while( GetMessage( &msg, NULL, 0, 0 ) ) {
    TranslateMessage( &msg );
    DispatchMessage( &msg );
}
return 0;
}
/**********  Windows Procedure  **********/
LRESULT CALLBACK MainWndProc( HWND hWnd, UINT msg, WPARAM wParam,    LPARAM
lParam )
{
PAINTSTRUCT ps;
HDC hDC;
TCHAR greeting[]=_T("Hello World!");
switch( msg ) {
case WM_PAINT:
        hDC = BeginPaint( hWnd, &ps );
       TextOut( hDC, 10, 10, greeting,_tcslen(greeting) );
        EndPaint( hWnd, &ps );
        break;
case WM_DESTROY:
    PostQuitMessage( 0 );
    break;
default:
     return( DefWindowProc( hWnd, msg, wParam, lParam ));
}
  return 0;
}
```

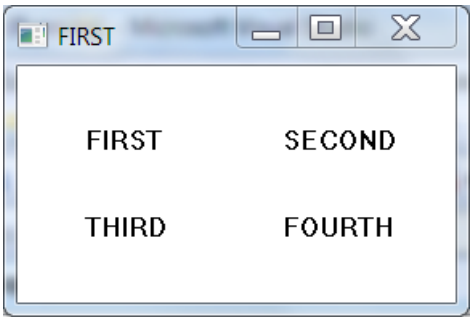
In visual studio, you can create a simple Win32 application by using an application wizard by choosing

File → New → Project → Win32 project

The program is modular and can be extended to add required application specific functionality.

Assignment 5: Writing Simple Widows program

1) Write a Simple Windows program with usual application window and modify it as follows. The cursor should be displayed as cross, the window color should be gray, the window caption should be “FIRST”. Display your name and details in the client area.
2) Write a Simple Windows program with usual application window. The window client area should be divided into four quadrants and write appropriately FIRST, SECOND, etc in the centre of each quadrant



Assignment Evaluation

0: Not Done [ ] 1: Incomplete [ ] 2: Late Complete [ ]

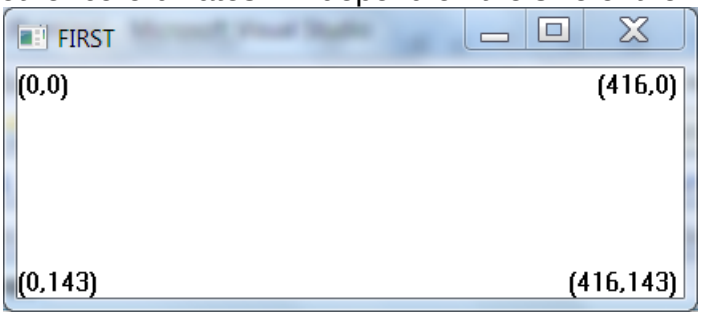
3: Needs Improvement [ ] 4: Complete [ ] 5: Well Done [ ]

Signature of the Instructor Date

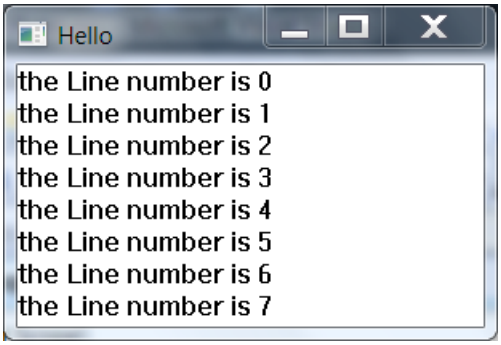
++

Assignment 6: Handling windows size message

3) Write a Simple Windows program with usual application window and display the co-ordinates of the four corners of the window. Ex The top left corner will be (0,0). The other co-ordinates will depend on the size of the window.



1) Write a Simple Windows program with usual application window and display n lines with the line numbers 0 to n to fill the complete client area. The number of lines should increase decrease when the size of the window is changed.



Assignment Evaluation

0: Not Done [ ] 1: Incomplete [ ] 2: Late Complete [ ]

3: Needs Improvement [ ] 4: Complete [ ] 5: Well Done [ ]

Signature of the Instructor Date

++

Assignment 7: Handling Keyboard messages

1) Write a Simple Windows program which displays in the client area the count of keystroke messages received of each type i.e WM_KEYUP, WM_KEYDOWN, WM_SYSKEYUP, WM_SYSKEYDOWN etc. The count should not change when the window looses the focus but should again start changing when the window regains the focus.

2) Write a Simple Windows program which displays in the client area the number of times each vowel (a, e, i, o, u) is pressed and also the consonants by processing WM_CHAR message. The count should not change if other than alphabet keys are pressed. The count should include both upper case as also lower case characters

3) Write a simple windows program which divides the client area into four quadrants and the caret can be moved between quadrants with the help of arrow keys. In the beginning the caret should be in top left quadrant.

For the above programs, students should handle keyboard messages. Any one program from the above set of programs need to be completed to complete the assignment

Assignment Evaluation

0: Not Done [ ] 1: Incomplete [ ] 2: Late Complete [ ]

3: Needs Improvement [ ] 4: Complete [ ] 5: Well Done [ ]

Signature of the Instructor Date

++

Assignment 8: Handling Mouse messages

1) Write a simple windows program that draws lines using rubber band technique. The mouse is dragged with left button down and for every movement of mouse the corresponding line from the start point(where the left button was pressed) to current positions are drawn and immediately replaced by new line when the mouse is moved. When the button is released the final line is displaced. The right button can be used for erasing the client area. Extend the program to include capturing of the mouse.

2) Write a simple windows program that displays the co-ordinates of the point as the mouse moves over the client area. When the left button is pressed the co-ordinates will be displayed at that point and the point selected will be numbered. More points can be similarly selected. Pressing right button will join all these points with lines.

3) Write a simple windows program which divides the client area into four quadrants and the mouse cursor can moved between quadrants. When the right mouse button is clicked it will display the quadrant name (first, second, third and fourth) at the current position.

For the above programs, students should handle mouse messages. Any one program from the above set of programs need to be completed to complete the assignment

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

Signature of the Instructor Date

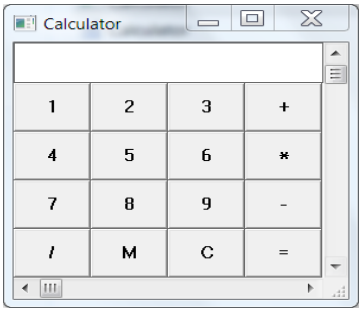
++

Windows (SDK) mini Project guidelines

The project is to be carried out with a team size of Two or Three. The SDK mini project should create a small utility tool.

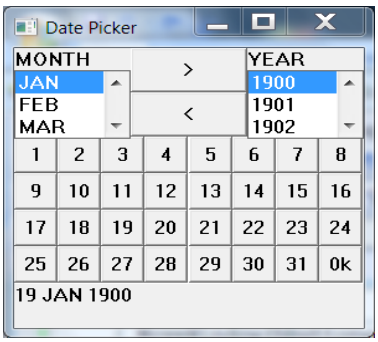
Some of the Project ideas are given below.

1) **Calculator program** = Write a Simple Windows program having the following functionalities of a calculator – The Interface has a text box, number and operator buttons apart from Memory ,Clear buttons and Equal buttons. The application has the following look.



The number can be typed in the text box directly or clicking number buttons, It can be followed by clicking the operator when text box clears out and next number can be typed followed by equal button to see the result in text box. Clear button can be used to clear the text box and Memory is toggle button. When clicked first time the contents of edit box are stored in Memory and on next use of Memory button the contents of memory are displayed in edit box. The program can be suitably extended to add two more functionalities of your choice.

2) **Date Picker -** Write a Simple Windows program having the following functionalities of a Date Picker – The Interface has a List box to select Month and another to select year showing ten values starting from 1900. Two buttons More and less can be used to change the entries in the year listbox by +10 or -10. For Selecting the date , 31 number buttons are displayed in four rows and 32nd button is ok. The selected date is displayed at the bottom after pressing ok. The program can be extended to have vertical scroll bar to the window, whose messages can be used to do what buttons more and less are doing



3) Write a program to create a simple editor using a multiline edit box with a menu containing following options
Count – displays the number of characters in a Messagebox
Upper – Converts all the characters in the edit box to upper case
Lower – Converts all the characters in the edit box to lower case
LineCount – displays the number of lines in a Messagebox
Cut – Removes selected text
Copy –Copies selected text to clipboard
Paste – Pastes the selected text at the caret position
WordCount- displays the number of words in a Messagebox
Insert – inserts 'Title of the file' as the first line in the text and
Inserts ' Name of the author' as the last line in the text
Delete – deletes the line at the current cursor position

4) Write a Windows program with a menu for selecting Pen color(Red Blue Green), different pen styles(solid ,dashed) and the points can be chosen by clicking the right button of the mouse, dragging the mouse and releasing the mouse at the second point and the line will be drawn with chosen pen style and color. Extend the Program to include menu for choosing different shapes(rectangle, pie, chord) and brush styles(solid ,hatch) and submenu for different hatch styles. The top corner of the bounding box can be chosen by clicking the right button of the mouse. The bottom corner can be chosen by dragging the mouse and releasing the mouse. The appropriate shape will be drawn filled with appropriate brush style.

5) Write a Windows program for playing a game of SUDOKU. The client area should display 9x9 textboxes. Some of them disabled and containing values. The user can add a digit in other textboxes. The menu can contain a validate item which when clicked checks whether there are duplications in row or column or individual nine 3x3 squares.

6) Write a Windows program for creating crossword puzzle. Provide a dialogbox for setting the number of rows and clolumns. The client area should display nxn textboxes. The user can right click on any box to disable and set the background to black and enter appropriate character in the remaining boxes. The menu item generate should generate a crossword puzzle for the same, numbering down and across values appropriately.

Introduction to Advanced Java

Advanced java course contains topics like database programming, Collection, servlets, JSP, socket programming. Basically Core java concepts are used to develop standalone applications. Advanced java concepts deals with client server architecture.

Database programming involves communicating with a database using Java program.

Collection framework provides different classes to collect the different types of data into a single unit.

Servlets and JSP can be used to develop web based applications. Client can specify the URL of servlet, JSP and they are executed at server side using tomcat web server.

Socket programming deals with intercommunication of programs running on different computers in the network. It introduces elements of network programming and concepts involved in creating network applications using Socket.

Assignments 9 to 13 are based on each topics that are mentioned above. These assignments will helps to get practical experience of Advanced Java concepts.

Assignment 14 is of case study. It contains a small real time application. To solve this assignment students need to use all techniques that they have studied in this course.

Assignment 9: Database Programming

Objectives

- **To communicate with a database using java.**
- **To execute queries on tables.**
- **To obtain information about the database and tables**

Ready Reference

JDBC : Java Database Connectivity

This API contains of a set of **classes** and **interfaces** to enable programmers to communicate with a database using java. These classes and interfaces are in the **java.sql** package.

The JDBC API makes it possible to do three things:

i. Load the required driver
ii. Establish a connection with a data source
iii. Send queries and update statements to the data source iv. Process the results

Load the driver

To load the driver, use the following command:

```
Class.forName("driverName");
```

Example: For postgre database

```
Class.forName("org.postgresql.Driver");
```

For this jdbc7.1-1.2 is required in student login. Specify the path of this file in .bashrc file of student login.

Establishing a connection

To establish a connection with the database, use the getConnection method of the DriverManager class. This method returns a Connection object.

```
DriverManager.getConnection("url", "user", "password");
```

Example:

```
Connection conn = DriverManager.getConnection
("jdbc:postgresql://192.168.100.4/TestDB", "scott", "tiger");
```

Executing Queries

To execute an SQL query, you have to use one of the following classes:

- Statement
- PreparedStatement

A Statement represents a general SQL statement without parameters. The method createStatement() creates a Statement object. A PreparedStatement represents a precompiled SQL statement, with or without parameters. The method prepareStatement(String sql) creates a PreparedStatement object.

Example:
ResultSet rs = stmt.executeQuery("SELECT * FROM Student");
int result = stmt.executeUpdate("Update authors SET name = 'abc' WHERE id = 1");

Process Result

ResultSet provides access to a table of data generated by executing a Statement. A ResultSet maintains a cursor pointing to its current row of data. The **next()** method is used to successively step through the rows of the tabular results.

Examples:
Statement stmt = conn.prepareStatement();
ResultSet rs = stmt.executeQuery("Select * from student");
while(rs.next())
{
//access resultset data
There are two forms of the getXXX methods:
i. Using columnName: getXXX(String columnName) ii. Using columnNumber: getXXX(int columnNumber)

Example
rs.getString("stuname");
rs.getString(1); //where name appears as column 1 in the ResultSet

Lab Assignments

Create a product table(id(primary key), name, price, quantity)

1. Write a menu driven program to perform the following operations on Product table.
   1. Insert (Generate product id by taking max(id) from table and increment by 1)
   2. Modify (Update product details by accepting id from user)
   3. Search
   4. View All
   5. Exit

2. Write a program to display information about all columns in the product table. (Use ResultSetMetaData).

3. Write a JDBC program to insert details of 5 products using batch update.

4. Write program to update price and quantity of all products by by 100 and 10 respectively. Display the details of products with modified price and quantity without closing connection in between update and display. (Use scrollable and Updatable ResultSet)

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

Signature of the Instructor Date

++

Assignment 10: Collections

Objectives

- **Study the Collections framework in java**
- **Use various collections**

Ready Reference

A collection is an object that represents a group of objects. A collection - sometimes called a container - is simply an object that groups multiple elements into a single unit. Collections are used to store, retrieve, manipulate, and communicate aggregate data. A collections framework is a unified architecture for representing and manipulating collections, allowing them to be manipulated independently of the details of their representation.

Implementations

The general-purpose implementations are summarized in the following table.

General-purpose Implementations					
Interfa	**Implementations**				
	Hash	**Resizable**	**Tree**	**Linked list**	**Hash table +**
Set	HashSet		TreeSet		LinkedHashSet
List		ArrayList		LinkedList	
Map	HashMa		TreeMa		LinkedHashMap

Iterator:

The **Iterator** interface provides methods using which we can traverse any collection. This interface is implemented by all collection classes.

Methods:

hasNext(	trueif there is a next element in the collection.
next()	Returns the next object.
remove()	Removes the most recent element that was

ListIterator

ListIteratoris implemented only by the classes that implement the List interface (ArrayList, LinkedList, and Vector).

hasNext()	trueif there is a next element in the collection
next()	Returns the next object
hasPrevious()	trueif there is a previous element in the
Previous()	Returns the previous object
remove()	Removes the most recent element that was

```
/* Program to demonstrate iterator */
import java.util.*;
public class
IteratorDemo
{
        public static void main(String[]
 args)
        {
                LinkedList d = new LinkedList();

                 d.add("A");
                 d.add("B");
                 d.add("C");
                 d.add("D");

                 ListIterator litr = d.listIterator();

                while(litr.hasNext())
                {
                        String elt = (String)litr.next();
                        System.out.println(elt);
                }

                System.out.println("Traversing Backwards : ");
                while(litr.hasPrevious())
                {
                         System.out.println(litr.previous());
                }
        }
}
```

Lab Assignments

1. Accept different n colors from user and store them in a collection. Display them in the sorted order. The collection should not accept duplicate elements. (Use a suitable collection). Display elements using iterator.

2. Create two singly linked lists for integer data. Perform following operations.
    a. Display both the list in reverse order
    b. Display intersection of two list
    c. Add element at first position in list1
    d. Remove the last element from list1

3. Create a Student table(rollno(primary key), name) in database. Write a JDBC program to fetch all student details and store them in Hashtable as key and value refers to rollno and name respectively. Search a student by accepting roll number from user. Display the name of the student if student is present otherwise display “Student not found”.

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

Signature of the Instructor Date

++

Assignment 11: Networking

Objectives

- **Introduction to the java.net package**
- **Connection oriented transmission – Stream Socket Class**

Ready Reference

Introduction

One of the important features of Java is its networking support. Java has classes that range from low-level TCP/IP connections to ones that provide instant access to resources on the World Wide Web. Java supports fundamental networking capabilities through java.net package.

/* Program to display socket information on client machine*/

Server.java

```
import java.net.*;
import java.io.*;
public class
Server
{
        public static void main(String args[])throws
        UnknownHostException, IOException
        {
                ServerSocket ss = new
                ServerSocket(4444);
                System.out.println("Server Started");
                Socket s = ss.accept();
                System.out.println("Connected to
                client");
          }
}
```

Client.java

```
import java.net.*;
import java.io.*;
public class Client
{
        public static void main(String args[]) throws
        UnknownHostException, IOException
        {
                Socket s = new Socket ("localhost",
                4444); System.out.println
                (s.getInetAddress()); System.out.println
                (s.getPort()); System.out.println
                (s.getLocalPort());
                s.close();
         }
 }
```

To run the program:

```
javac Server.java
javac Client.java
java Server
java Client
```

Lab Assignments

1. Write a program to display Host name and Host address of machine. (Use InetAddress class)

2. Write a server program which echoes messages sent by the client. The process continues till the client types “END”.

3. Write a program which sends the name of a text file from the client to server and displays the contents of the file on the client machine. If the file is not found, display an error message.

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

Signature of the Instructor Date

++

Assignment 12: Servlets

Objectives

- **To understand server-side programming**
- **Defining and executing servlets**

Ready Reference

Servlets are small programs that execute on the server side. Servlets are pieces of Java source code that add functionality to a web server.

Running servlets requires a server that supports the technologies. Several web servers, each of which has its own installation, security and administration procedures, support Servlets. The most popular one is the Tomcat- an open source server developed by the Apache Software Foundation in cooperation with Sun Microsystems version 5.5 of Tomcat supports Java Servlet.

Writing, Compiling and Running Servlet

SimpleSerlvet.java

```
/* Program for simple servlet*/
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class SimpleServlet extends HttpServlet
{
        public void service(HttpServletRequest req,
        HttpServletResponse res) throws ServletException,
        IOException
        {
                res.setContentType("text/htm
                l"); PrintWriter
                out=rs.getWriter();
                out.println("<B>Hello, Welcome to Java
                Servlet's</B>");
                out.close();
         }
}
```

1. Write above SimpleServlet.java program. Compile it with the Java compiler as: javac SimpleServlet.java. After compilation it creates SimpleServlet.class.
2. To make the servlet available, you have to publish this class file in a folder on your web server that has been designated for Java servlets. Tomcat provides the **classes** sub-folder to deploy this servlet's class file. Copy this class file in this classes sub-folder, which is available on

the path: tomcat/webapps/ WEB-INF/classes.

3. Now edit the **web.xml** file available under WEB-INF sub-folder with
4. the following lines:

```
<servlet>
    <servlet-name>SimpleServlet</servlet-name>
    <servlet-class>SimpleServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>SimpleServlet</servlet-name>
    <url-pattern>/SimpleServlet</url-pattern>
</servlet-mapping>
```

4. Repeat the above sequence of line to run every newly created servlet. Remember, these line lines must be placed somewhere after the <web-app> tag and before the closing </web-app> tag.

5. After adding these lines, save web.xml file. Restart the Tomcat service and run the servlet by loading its address with a web browser as: http://ipaddress of tomcatserver:port of tomcat/SimpleServlet.

Using Postgre – Database Connectivity tool with servlets

Java's Servlet also provides support for data handling using Postgre database. For this Copy jdbc7.1-1.2 file into the subfolder: tomcat/lib

Lab Assignments

1. Write a servlet program to display current date and time of server.

2. Design a servlet to display "Welcome IP address of client" to first time visitor. Display "Welcome-back IP address of client" to repeat visitor. (Use Cookies)
   (Hint: Use req.getRemoteAddr() to get IP address of client)

3. Write a servlet program to create a shopping mall. User must be allowed to do purchase from two pages. Each page should have a page total. The third page should display a bill, which consists of a page total of whatever the purchase has been done and print the total. (Use HttpSession)

4. Design the table User(login_name, password) using Postgre. Also design an HTML login screen. Accept the login name and password from the user. Write a servlet program to accept the login name and password and validates it from the database you have created. If it is correct then display welcome.html otherwise display error.html.

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

Signature of the Instructor Date

++

Assignment 13: Java Server Pages□ □ □ □ □ □

Objectives

- **To demonstrate the use of JSP**

Ready Reference

JSP is Java Server Page, which is a dynamic web page and used to build dynamic websites. To run jsp, we need web server which can be tomcat provided by apache, it can also be jBoss(Redhat), weblogic (BEA) , or websphere(IBM).

To run JSP files: all JSP code should be copied (Deployed) into tomcat/webapps folder.
To execute the file, type: http://server-ip:portno/filename.jsp

```
<html>
        <body>
        <% out.print("Hello World"); %>
        </body>
</html>
```

Lab Assignments

1. Write a JSP program to perform Arithmetic operations such as Addition, Subtraction, Multiplication and Division. Design a HTML to accept two numbers in text box and radio buttons to display operations. On submit display result as per the selected operation on next page using JSP.

2. Create a JSP page, which accepts user name in a text box and greets the user according to the time on server side.
   Example: If user name is Admin
   Output:
   If it is morning then display message in **red** color as,
   Good morning Admin
   Today's date is : dd/mm/yyyy format
   Current time is : hh:mm:ss format

   If it is afternoon then display message in **green** color as,
   Good afternoon Admin
   Today's date is : dd/mm/yyyy format
   Current time is : hh:mm:ss format

   If it is evening then display message in **blue** color as,

Good evening Admin
Today's date is : dd/mm/yyyy format
Current time is : hh:mm:ss format

(Hint: To display date and time use GregorianCalendar and Calendar class)

3. Write a JSP program "weekdays.jsp" to display all weekdays from Monday to Sunday using static variable. Declare a static variable as count using declaration tag. Assign count as 1. Also add hyperlink as "Click Me". Assign href="weekdays.jsp".
   Consider the following scenario to run JSP program.
   a. Initially count is 1 so display "Monday" and increment a count by 1.
   b. Click on "Click Me", now count is 2 so it will display "Tuesday" and increment a count by 1.

   This way every time when we click on "Click Me" it should display weekdays starting from Monday to Sunday. Once it reaches to Sunday then again restart the count by assigning 1.

4. Write a JSP program to display details of products from database in tabular format. Use Product table from database assignment.

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

Signature of the Instructor Date

++

Assignment 14: Case Study

Design a following tables in database.

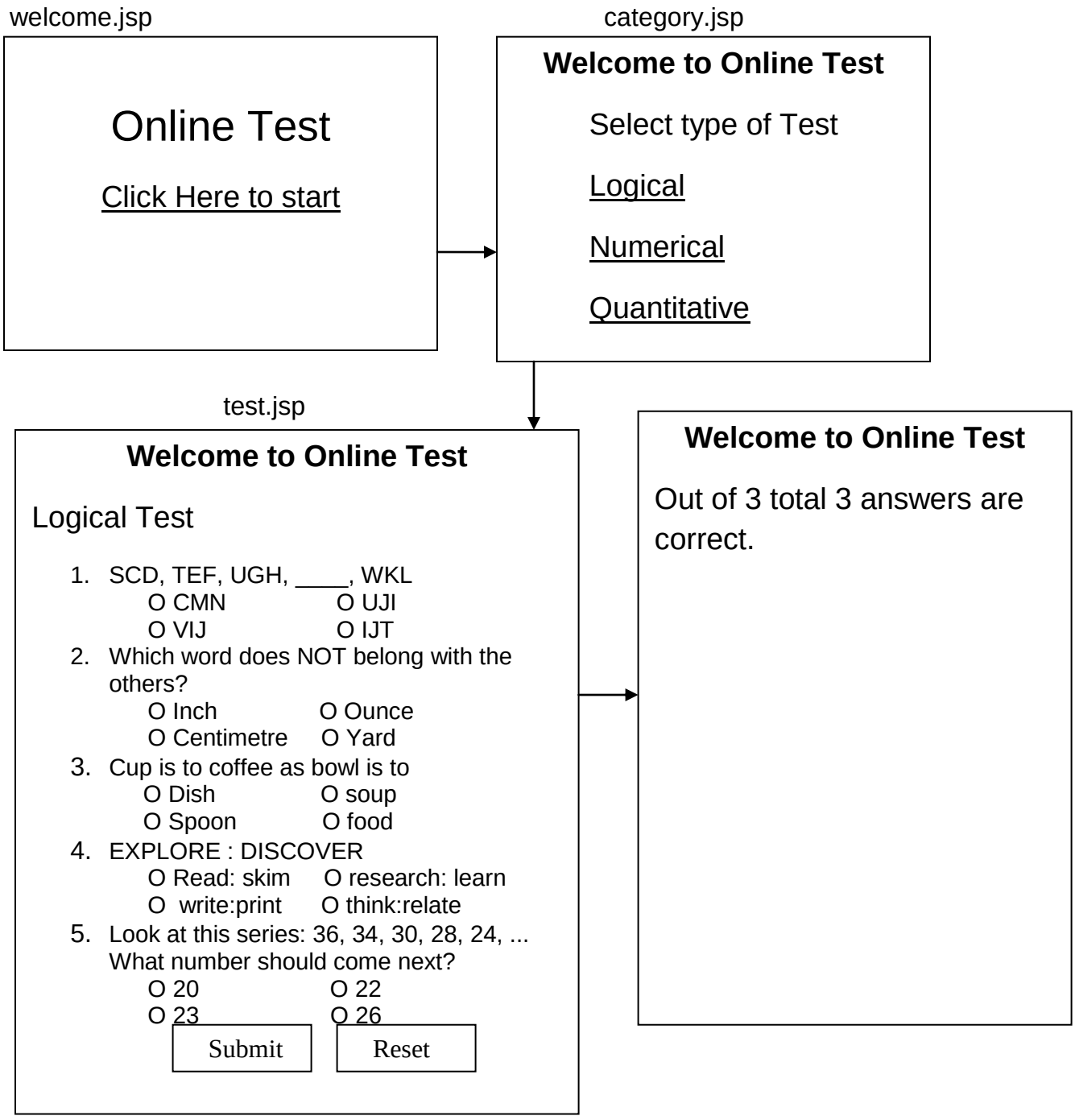
1. test_type

   tid(primary key), cname

2. Question

   qid(primary key), ques, op1, op2, op3, op4, answer, tid(foreign key of test_type)

   [Add 5 questions of each test type]



Use the concept of database, Servlet, JSP and collection for designing Online Test. Initially display “welcome.jsp”. After clicking on hyperlink display types of test from database. After selecting test type display question and options from database. On Submit display the result of total correct answers.

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

Signature of the Instructor Date

++

Bibliography

Computer Graphics

Computer Graphics using OpenGL , 3rd Edition By F. S. Hill , Stephen M. Kelly

SDK

Programming Windows®, Fifth Edition, by Charles Petzold, Microsoft

Advanced Java

Complete reference Java by Herbert Schildt(5th edition)

Core Java Volume-II-Advanced Features, Eighth Edition, Cay S. Horstmann, Gary Cornell, Prentice Hall, Sun Microsystems Press.