

SMART WAY TO LEARNING C TUTOR

A Project Work submitted in partial fulfillment of the

requirements for the degree of

BACHELOR COMPUTER APPLICATIONS

to the

Periyar University, Salem-11

Submitted By

S. TAMILARASAN

REG No.: C22UG132CAP026



DEPARTMENT OF COMPUTER APPLICATIONS

JAIRAM ARTS & SCIENCE COLLEGE

(AFFILIATED TO PERIYAR UNIVERSITY)

CHINNATHIRUPATHI, SALEM – 08

(MARCH/APRIL - 2025)

CERTIFICATE

MR. S. SUNDHARAM, MSc., M.Phil.,

DATE:

ASSISTANT PROFESSOR OF
COMPUTER SCIENCE,
JAIRAM ARTS AND SCIENCE COLLEGE,
SALEM-08.

This is to certify that the Project Work entitled "**SMART WAY TO LEARNING C TUTOR**" submitted in partial fulfillment of the requirements of the degree of Bachelor of Computer Application to the Periyar University. Salem is a record of bonafide work carried out by S **TAMILARASAN** Reg. No. **C22UG132CAP026** under my supervision and guidance.

Head of the Department

Internal Guide

Date of Viva-voice:

Internal Examiner

External Examiner

ACKNOWLEDGEMENT

ACKNOWLEDGEMENT

At the outset I would like to express my first and the foremost gratitude to the “Almighty”, for the merciful shower of grace and blessing in all the endeavors to bring out the project successful.

I take this opportunity to express my sincere thanks and whole hearted gratitude to my revered and beloved Chairman Mr. J. RAJENDRA PRASAD

I take this opportunity to express my sincere thanks and whole hearted gratitude to my revered and beloved Correspondent Mr.R.MANIKANDAN, M.COM.,M.PHIL

I take this opportunity to express my deep sense of gratitude and profound thanks to Principal Dr.K.PALANISAMY, M.C.A.,M.Phil.,Ph.D., for all blessings and help to provided during the project period.

I proudly thanks to Mr. J.KARTHICK, MCA., Head of the Department, UG Computer Application for providing a great opportunity to pursue this project work.

I extremely grateful to my guide Mr. S. SUNDHARAM, MSc., M.Phil., Assistant professor. UG Department of Computer Science. His constant encouragement, valuable suggestions, continuous help and creative ideas throughout the period of the project work.

I also express my sincere thanks to All Teaching and Non-Teaching Staffs in UG Department of Computer Applications for their well support throughout my project work.

Finally I also grateful to my Friends they are inspired me all through the days of my thesis work. Last but not least, I thank everybody who was a source of help to complete this project work effectively.

S TAMILARASAN

CONTENTS

CONTENT

S NO	INDEX	PAGE NO
	ABSTRACT	
1.	1.1 INTRODUCTION	2
2.	1.2 SYSTEM SPECIFICATION 1.2.1 HARDWARE CONFIGURATION 1.2.2 SOFTWARE CONFIGURATION	3
3.	SYSTEM STUDY 2.1 EXISTING SYSTEM 2.1.1 DRAWBACKS 2.1.2 PROPOSED SYSTEM 2.1.3 FEATURES	4
4.	SYSTEM DESIGN & DEVELOPMENT 3.1 FILE DESIGN 3.2 INPUT DESIGN 3.3 OUTPUT DESIGN 3.4 DATABASE DESIGN 3.5 SYSTEM DEVELOPMENT 3.6 DESCRIPTION OF MODULES	8
5.	TESTING AND IMPLEMENTATION	17
6.	CONCLUSION	20
7.	BIBLIOGRAPHY	21
8.	APPENDICES A.DATA FLOW DIAGRAM B.TABLE STRUCTURE C.SAMPLE CODING D. SAMPLE INPUT AND OUTPUT	22

SYNOPSIS

SYNOPSIS

The C Tutor Android Application is to satisfy or fulfill the student's need, which helps them to learn C programming language easily and effectively. It is an application that allows students to develop a program in c and prepare for the interview without consuming much time. Since C programming is the basic of all programs, through our application Students can learn the flow as well as other basic needs that help him/her to write an effective program in C language.

The design of such an application is made full of Java and Android code. This application is mainly developed to help the students for learning the basics that will help them to develop an effective program in C as well as other programming language Since C programming is the basic for all other programming languages. Even this helps them to prepare for an interview with the frequently asked questions given in our application. Our application helps the staff members to refer the important concepts and programs of C programming that helps them to reduce the time they spend by referring many books. So they can teach effectively without referring any books. In our application we include an option called frequently asked interview questions in C that help the final year students to prepare for technical interview without any tension .Some website references for c-programming and preparation for Interview also given for the benefit of students(this feature will require internet connection). And in future our application will be upgraded with various features.

INTRODUCTION

1. INTRODUCTION

1.1 PROJECT DESCRIPTION

Android, the world's most popular mobile platform. Android powers hundreds of millions of mobile devices in more than 190 countries around the world. Android gives us world-class platform for creating apps and games for android users.

Android gives us everything that we need to built best-in-class apps experience. It provides us tools for creating our own apps that look great and take advantage of the hardware capabilities available on each device.

To help us develop an Android app efficiently, the Android developer tool offer a full Java IDE with advanced features and also helps in debugging and packaging Android apps. Google Play is the premier market place for selling and distributing Android apps.

SYSTEM REQUIREMENTS AND SPECIFICATION

1.2 SYSTEM REQUIREMENTS AND SPECIFICATION

1.1.2 HARDWARE REQUIREMENTS

- Processor : Intel Pentium or Higher
- Memory : 2 GB
- Hard Disk Space : 500 GB

1.1.3 SOFTWARE REQUIREMENTS FOR CLIENT

- Android Smart Phone

1.1.4 SOFTWARE REQUIREMENTS FOR SERVER

- ADT Bundle
- Java Virtual Machine 1.6 or higher
- Dalvik Virtual Machine
- Emulator

SYSTEM STUDY

2. SYSTEM STUDY

2.1 EXISTING SYSTEM

The existing system is manual. Students and Staffs have to take book from library or buying them to refer even a small concept. Even though internet is available for referring purpose it may require us to spend money for availing internet. This case becomes even worse because of connectivity problems.

2.1.1 DRAWBACKS

- Basic concepts of providing our applications.
- Doesn't explain the entire concept.
- There is no runtime checking.
- There is no strict type checking.
- This application doesn't have the concept of constructor or destructor.

2.2 PROPOSED SYSTEM

There are so many reasons for introducing the new system. Before, it was man driven and lengthy process. During the process of referring manually to the books from library, if there is no library or any shops available in our area or we want to refer a small concept which is important or we need some reference of a program or some tutorial sites immediately, here where our proposed system plays an important role. Hence the proposed system will help in saving time, always handy and easy to access.

The advantages of the proposed system are as follows:

- User interactive
- Effective
- Error-free
- Reduces time consuming process
- Minimize the user's work
- Always handy

2.3 FEATURES

- In future our application can be upgraded with various features.
- We are planning to include a quiz module that will help the users to check their ability for study and interview purpose.
- We will provide periodical updates on interview questions that will help the students with updated knowledge for attending the interview.
- We are also planning to provide a module to develop user's general awareness too.

- Not only will our app be helpful for students it can be helpful for every user that install our application in their smart phone when our application is uploaded in the Google play store.

2.4 SOFTWARE DESCRIPTION

2.4.1. ANDROID

Android is a free, open source mobile platform. It is a Linux-based, multiprocessing, multithreaded OS. Android is not a device or not a product. The android software stack is, put simply, a Linux kernel and a collection pf C/C++ libraries exposed through an application framework that provides services for, management of, the run time and applications.

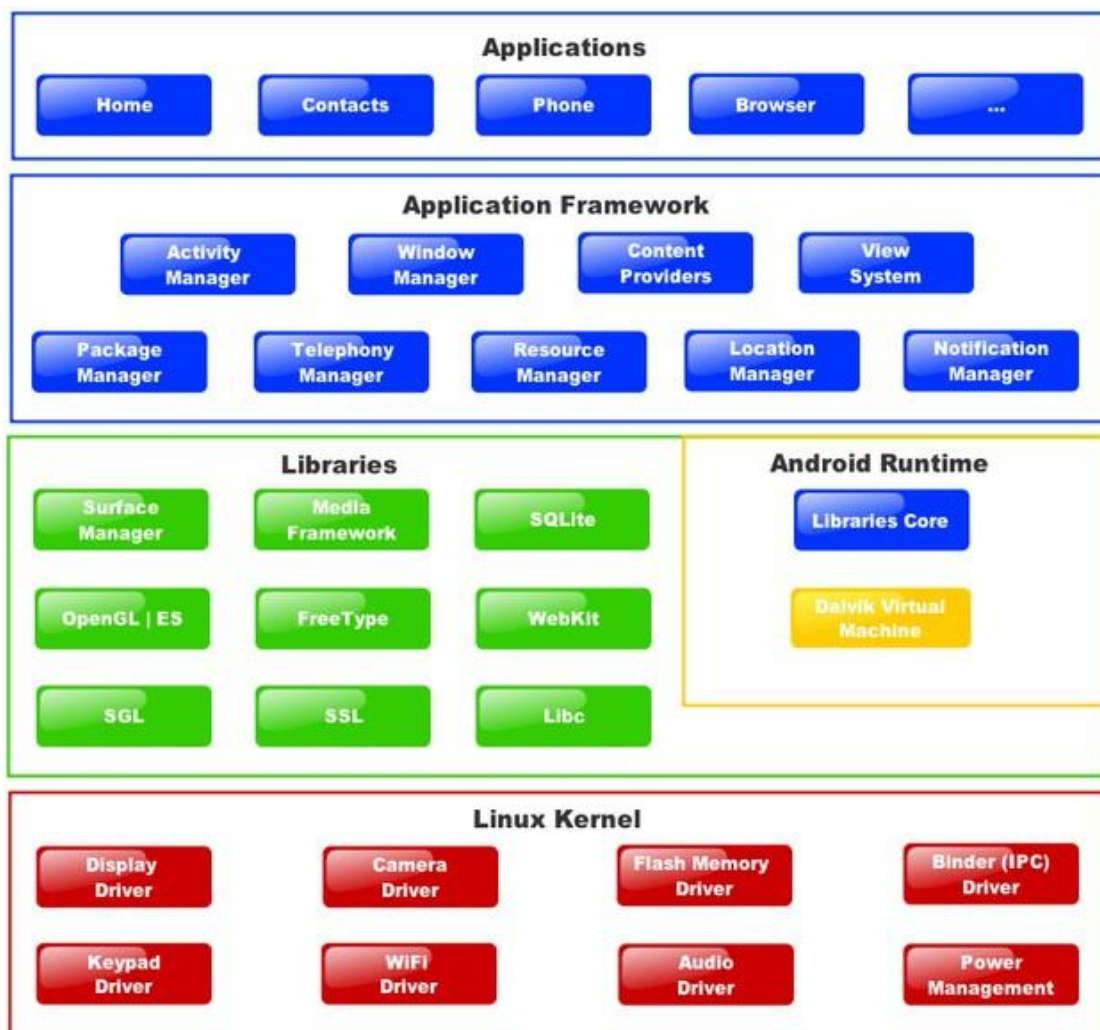


Fig Android Architecture

Linux Kernel: Core services are handled by a Linux 2.6 kernel. The kernel also provides an abstraction layer between the hardware and the remainder of the stack.

Libraries: Running on top of the kernel, Android includes various C/C++ core libraries such as libc and SSL, as well as the following:

- A media library for playback of audio and video media

- A surface manager to provide display management
- Graphics libraries that include SGL and OpenGL for 2D and 3D graphics
- SQLite for native database support
- SSL and WebKit for integrated web browser and internet security

Android Runtime: The run time is what makes an android phone rather than a mobile Linux implementation. Including the core libraries and the Dalvik VM, the Android run time is the engine that powers our applications and, along with the libraries, forms the basis for the application framework.

- **Core libraries** – Although most Android application development is written using the Java language, Dalvik is not a Java VM. The core Android libraries provide most of the functionalities available in the core Java libraries, as well as the Android specific libraries.
- **Dalvik VM** – Dalvik is a register-based Virtual Machine that's been optimized to ensure that a device can run multiple instances efficiently. It relies on the Linux kernel for threading and low-level memory management.

Application Framework: The application framework provides the classes used to create Android applications. It also provides a generic abstraction for hardware access and manages the user interface and application resources.

Application Layer: All applications, both native and third-party, are built on the application layer by means of the same API libraries. The application layer runs within the Android run time, using the classes and services made available from the application framework.

JAVA VIRTUAL MACHINE

A java virtual machine is a process virtual machine that can execute Java byte code. It is the code execution component of the java platform. JVM interprets compiled java binary code (called byte code) for a computer processor so that it can perform a java programs instruction. Java was designed to allow application program to be built that could be run on any platform without having to be rewritten or recompiled by the programmer for the each separate platform. JVM makes this possible because it is aware of the specific instruction lengths and other particularities of the platform.

DALVIK VIRTUAL MACHINE

Dalvik is the process virtual machine in Google's android operating system. It is the software that runs the apps on android device. Dalvik is thus an integral part of android, which is typically used on mobile devices such as mobile phones and tablet computers as well as more recently on embedded devices such as smart TVS and media streamers. Programs are commonly written in java and

compiled to bytecode. They are then converted from java virtual machine compatible java class files to Dalvik compatible .dex (dalvik executable) and odex (optimized dalvik executable) files before installation on a device, giving rise to the related terms odexing and de-odexing. The compact dalvik executable format is designed to be suitable for systems that are constrained in terms of memory and processor speed. Dalvik is open source software.

ARCHITECTURE

Unlike java VMs, which are stack machines, the Dalvik VM uses registered base architecture. A tool called dx is used to convert from some java .class files into the .dex format. Multiple classes are included in a single .dex file. Duplicate strings and other constants used in multiple class files are included only once in the .dex output to conserve space. Java byte code is also converted into an alternative instruction set used by the dalvik VM. An uncompressed .dex file is typically a few percent smaller in size than a compressed .jar derived from the .class files.

EMULATOR

Atruntime, the emulator reads and writes data to 2 disk images: a user data image and optionally an SDK card image theses images emulate the user-data partition and removal storage media on natural device. The emulator provides a default user–data disk image. At startup, the emulator creates a default image as a copy of the system user-data image. The emulator stores the new image with files of the active AVD. The emulator provides startup option to let us override the actual names and storage locations of the runtime images to load.

SYSTEM DESIGN AND DEVELOPMENT

3. SYSTEM DESIGN AND DEVELOPMENT

Design is the process of translating requirements defined during analysis into several designs activities for user requirements. The designer select requirements needed to implement the system in this phase; the design of the database also takes place.

After identifying the problem, limitations are opportunities to improve the efficiency system. A detail design of the proposed system is done. In database design several objectives are considered such as,

- Controlled Redundancy
- Data Independence
- More Information at low cost
- Accuracy and Integrity
- Recovery and Failure
- Security
- Performance

3.1 FILE DESIGN

The J2EE platform simplifies enterprise applications by basing them on standardized, modular components, by providing a complete set of services to those components, and by handling many details of application behavior automatically, without complex programming.

3.2 INPUT DESIGN

Input design is the process of converting user oriented inputs to a computer based format. The quality of the system input determines the quality of system output. Input design determines the format and validation criteria for data entering to the system.

Input design is a part of the overall system design, which requires very careful attention. If the data going into the system is incorrect then the processing and output will magnify these errors. Input can be categorized as internal, external, operational, computerized and interactive. The analysis phase should consider the impact of the inputs on the system as a whole and on the other systems.

The main objectives considered during input design are nature of input processing, Flexibility and thoroughness of validation rules, Handling of priorities within the input Documents, Screen design to ensure accuracy and efficiency of input Relationship with files. Careful design of the input also involves attention to error handling, controls, batching and validation procedures. Here the inputs are designed is such a way that occurrence of errors are minimized to its maximum.

The inputs entered by the customers are checked at the client and then fielded to the Database. Any abnormality found in the inputs are checked and handled effectively. Input design features can ensure the reliability of a system and produce results from accurate data or they can result in the production of erroneous information.

The features of input screen are: Well defined messages and prompts, Clear labels form menu items and fields, Clutter free screen.

3.3 OUTPUT DESIGN

Output generally refers to the results and information that are generated by the system. For many end users, an output is the aim reason for developing the system and the basis on which they will evaluate the usefulness of the application. Most end-users will noticeably operate the information system or enter data through workstation, but they will use the output from the system.

When designing output, system analyst must accomplish the following:

- Determine what Information to present decide
- Where to display, print the information and
- Select the output medium

3.4 DATABASE DESIGN

Database design is part of the database development process that involves analysis of a problem definition (specifications and requirements) and provides all necessary findings for building a logical structure of data. The problem definition specifies more or less formally the purpose, needs, requirements and constraints for data expected to support some organizational operations. The logical structure of data may initially be expressed in a plain language. It may, for example, consist of a series of simple statements (subject + predicate + object) that can be transformed into a more expressive data model. Such a model comes close to what one considers a conceptual level or view level and it should allow for precise mapping to a database schema (metadata).

3.5 MODULE DESCRIPTION

We categorize our C-Tutor application in to four modules:

- Web View
- Tutorial
- List of Programs
- Interview questions (FAQ)

Web View

Students can access the some important and informative websites through C-tutor application with internet connection. These sites include some important programming sites and some sites that will help students in the preparation of interview. The main advantage of accessing the website through C-tutor application is to avoid accessing the website through search engines such as Google, yahoo.,etc. This is done using the concept called WebView. It uses the Web Kit rendering engine to display web pages and includes methods to navigate forward and backward through a history, zoom in and out, and perform text searches and more.

In order for an Activity to access the internet and load web pages in a WebView, should add the INTERNET permissions to Android Manifest file:

<uses-permission android: name="android.permission.INTERNET">

A WebView has several customization points where students can add their own behavior. Those are:

- Creating and setting a WebChromeClient subclass. This class is called when something that might impact a browser UI happens.
- Creating and setting a WebViewClient subclass. It will be called when things happen that impact the rendering of the content.
- Modifying the WebSettings, such as enabling JavaScript with setJavaScriptEnabled ().
- Injecting java objects into the WebView using the addJavascriptInterface (Object, String) method. This method allows students to inject java objects into a page's JavaScript context, so that they can be accessed by JavaScript in the page.
- To enable the built-in zoom, set WebSettings.setBuiltZoomControls (Boolean). Using zoom if either the height or width is set to WRAP_CONTENT may lead to undefined behavior and should be avoided.
- In order to support inline HTML5 video in student's application, students need to have hardware acceleration turned on, and set a WebChromeClient.

Tutorial

This module provides some important concepts in C-programming language that will help both staffs and students to refer some very important concepts from anywhere and anytime. The titles of the concepts were provided neatly in our application in the way that users can access our application easily. We provided the titles as buttons those were neatly arranged one by one in the scrollview which will help user to scroll and select the concept he want from the list of concepts provided. The contents for the titles were also provided neatly in the way that will help the user to understand the concepts quickly and easily. We provided every content in a file that will not be disturbed when touching or scrolling done by user since the file is not editable the file is first copied to raw folder and then loaded

to the application on a call to the file name in xml file of the activity that requires file. The file is first read and then written to the application as a resource or asset that cannot be compilable as well as uneditable. The code we used for reading and writing is as follows,

```
InputStream inputStream = getResources().openRawResource(R.raw.hello);
    ByteArrayOutputStream byteArrayOutputStream = new ByteArrayOutputStream();
    int i;
    try {
        i = inputStream.read();
        while (i != -1)
        {
            byteArrayOutputStream.write(i);
            i = inputStream.read();
        }
    }
```

List of programs

This module provides the user with important programs that will help in their provisional as well as professional career. This will help the user to refer important programs they want anywhere and anytime without the requirement of any other resources such as books or internet. All you need is an Android mobile with our application in it. Like tutorial module this module also arranged neatly with title as buttons in scrollview. This helps the user to easily refer the program he/she wants. The programs were loaded in files like previous module which in turn loaded into the application. Users have to click on title of the program to see the flow and output of the program. Explanation of the program also provided for easy understanding of the programs.

Interview Questions (FAQ)

In this module, we have provided the frequently asked interview questions in C-programming language. This helps students to know what a company need from the student in technical aspect and what areas that the student should concentrate to attend the technical round of an interview. This App acts as handy, that they can easily and quickly refer to some important concepts and tricky questions for the interview purpose. We have integrated it using the concept of ScrollView, so that users can easily access the module and search for the questions for the companies he/she wants. We provided this functionality using webview, in which the questions for selected companies can be opened using the browser.

Components used for Modules

We use the following components to build our Application's module:

- Activity
- Intent
- Layout
- View

The description of the above components is as follows,

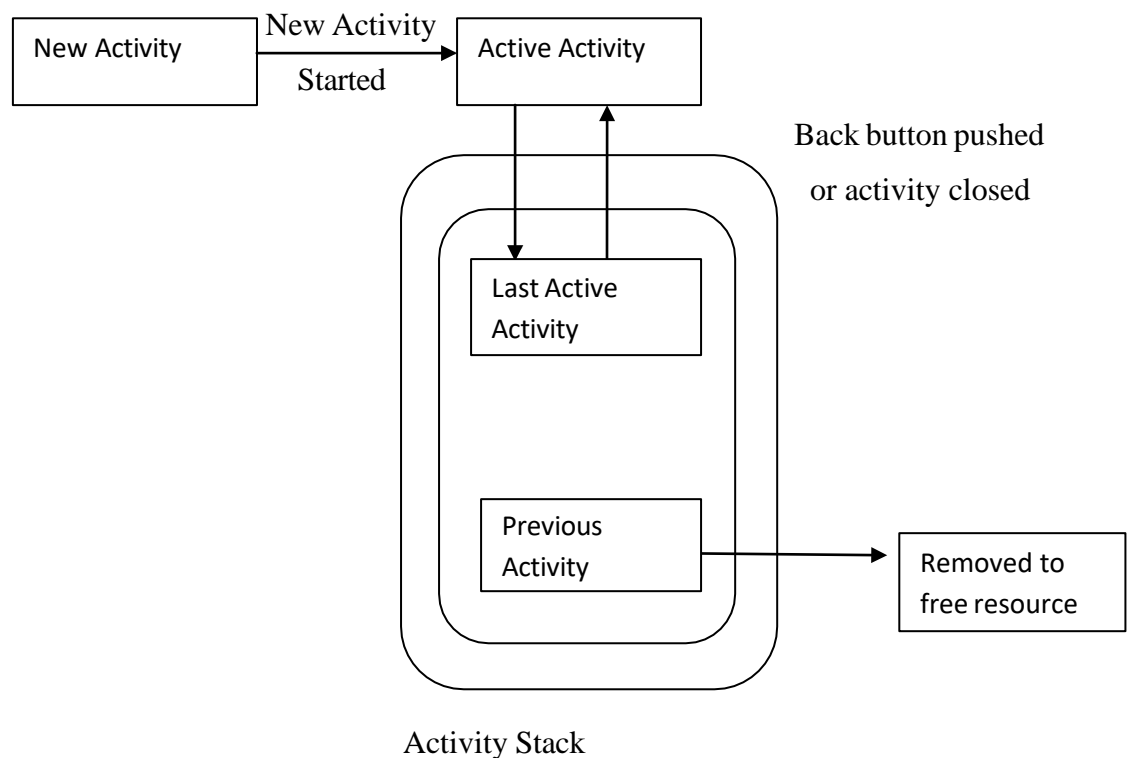
Activity

Each Activity represents a screen that an application can present to its users. The more complicated our application, the more screens we are likely to need. Typically, this includes at least a primary interfaces screen that handles the main UI functionality of our application. This primary interface generally consists of a number of fragments that make up our UI and is generally supported by a set of secondary Activities. To move between screens we start a new Activity (or return from one). Most Activities are designed to occupy the entire display, but we can also create semitransparent or floating Activities.

The state of each Activity is determined by its position on the Activity stack, a last-in-first-out collection of all the currently running Activities. When a new Activity starts, it becomes active and it's moved to the top of the stack.

As Activities are created and destroyed, they move in and out of the stack, there are four possible states:

- **Active** – when an Activity is at the top of the stack it is visible, focused, foreground Activity that is receiving user input. Android attempt to keep it alive at all costs, killing Activities further down the stack as needed, to ensure that it has the resources it needs. When another Activity becomes active, this one will be paused.



(FigActivity)

As Activities are created and destroyed, they move in and out of the stack, there are four possible states:

- **Active** – when an Activity is at the top of the stack it is visible, focused, foreground Activity that is receiving user input. Android attempt to keep it alive at all costs, killing Activities further down the stack as needed, to ensure that it has the resources it needs. When another Activity becomes active, this one will be paused.
- **Paused** – in some cases our Activity will be visible but will not have focus, at this point it's paused. This state is reached if a transparent or non-full-screen Activity is active in front of it. When paused, an Activity is treated as if it were active; however, it doesn't receive user input events.
- **Stopped** – when an Activity isn't visible, it "stops". The Activity remains in memory, retaining all state information.
- **Inactive** – After an Activity has been killed, and before it's been launched, it's inactive. Inactive Activities have been removed from Activity stack and need to be restarted before they can be displayed and used.

Intent

Intents are asynchronous messages which allow application components to request functionality from other Android components. Intents allow you to interact with components from the same applications as well as with components contributed by other applications. For example, an activity can start an external activity for taking a picture.

Intents are objects of the `android.content.Intent` type. Your code can send them to the Android system defining the components you are targeting. For example, via the `startActivity()` method you can define that the intent should be used to start an activity.

An intent can contain data via a `Bundle`. This data can be used by the receiving component. Activities which are started by other Android activities are called *sub-activities*. This wording makes it easier to describe which activity is meant. We can also start services via intents. Use the `startService(Intent)` method call for that.

View

The basic building block for user interface is a View object which is created from the View class and occupies a rectangular area on the screen and is responsible for drawing and event handling. View is the base class for widgets, which are used to create interactive UI components like buttons, text fields, etc.

The ViewGroup is a subclass of View and provides invisible container that hold other Views or other ViewGroups and define their layout properties. At third level we have different layouts which are subclasses of ViewGroup class and a typical layout defines the visual structure for an Android user interface and can be created either at run time using View/ViewGroup objects or you can declare your layout using simple XML file `main_layout.xml` which is located in the `res/layout` folder of your project.

This tutorial is more about creating your GUI based on layouts defined in XML file. A layout may contain any type of widgets such as, and so many views they can.

- Buttons
- EditTexts
- TextViews
- Labels

Layout

Android Layout is the most interesting View element container in Android. To design UI for an App, you need to know about Android Layout. It is also known as **Viewgroup**. All view elements like textbox, label, buttons etc need to be contained inside the layout. This Layout is in-turn contained in a layout file location inside **res/layout** directory of the Android app. An Android Activity uses the Android Layout & shows output on the screen.

The two most important kind of Android Layout elements are

- LinearLayout
- RelativeLayout

Android Linear Layout

It is one of the fundamental layout in Android which is available for developers to implement their UI's. As the name indicates, this layout is "linear" in the sense that it puts its children in a linear fashion. LinearLayout is a view group that aligns all its children in a single direction, vertically or horizontally.

So, the available orientations for LinearLayout are :

- Vertical
- Horizontal

A Sample code for linear layout is as follows,

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <!-- Other views -->
</LinearLayout>
```

Relative Layout

Relative layout is one of the basic layouts available to design UI in Android. It arranges its child views with reference to the view siblings or with respect to itself. That is, using RelativeLayout you can position a view to be **toLeftOf**, **toRightOf**, **below** or **above** its siblings. You can also position a view with respect to its parent (centered horizontally, vertically or both, or aligned with any of the edges of the parent RelativeLayout).

In this layout, you can specify the child layouts w.r.t its sibling or parent. One or more of the following attributes must be set for every view. Otherwise, the views will be rendered one over another in the **top left** region of the RelativeLayout.

A sample code for relative layout is as follows,

```
<?xmlversion="1.0"encoding="utf-8"?>
<RelativeLayoutxmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">
    //Other views here
</RelativeLayout>
```

TESTING AND IMPLEMENTATION

4. TESTING AND IMPLEMENTATION

System Testing

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product it is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

TYPES OF TESTS

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.
- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results:

All the test cases mentioned above passed successfully. No defects encountered.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results:

All the test cases mentioned above passed successfully. No defects encountered.

Implementation

Our team can provide a Maintenance Plan that covers common and general updates an App may require. As the mobile platforms are evolving very quickly, it's not uncommon for an App to need some adaption or modification when new OS versions are released. As part of our Maintenance Plans, we can receive the necessary reviews, changes and republishing of our project as needed to stay compatible with the changing landscape of OS versions. Also, some apps require regular maintenance of databases, feeds, links and other content which may be only partially dynamic or externally modifiable.

CONCLUSION

CONCLUSION

We hereby conclude that our application will satisfy the students need in such a way that they can understand the important concepts and programs in C-programming language and will help them in preparation of interview. The programming examples provided in our application will help the users to learn about flow and error control while they start programming. And also our application helps staffs to refer important concepts and programs from anywhere and anytime. Our application will play a vital role in students development, their performance, etc.

BIBLIOGRAPHY

BIBLIOGRAPHY

REFERENCE BOOKS

1. RetoMerier, “Professional Android Application Development”
Wiley Publication, Inc. Edition 2014
2. MarkoGargenta, “Learn Android”, O’Reilly Media Edition 2011
3. AllBharami, “ObjectOrientedSystemDevelopment”,
Tata McGraw –Hill Edition 2008.

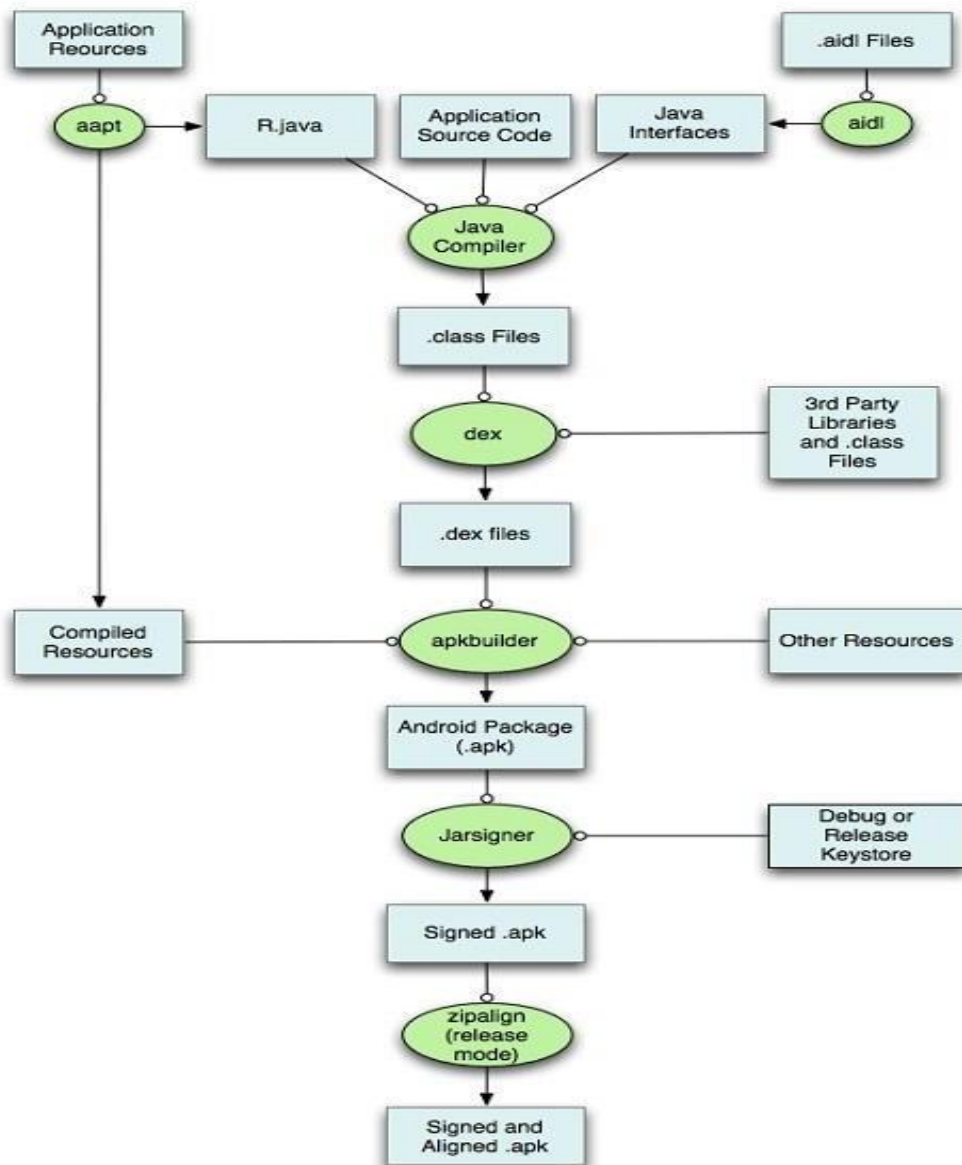
WEB REFERENCES

1. www.androidexample.com
2. www.coderzHeaven.com
3. www.examples.javacodegeeks.com
4. www.androidhive.info
5. www.tutorialspoint.com
6. www.freashersworld.com
7. www.stackoverflow.com

APPENDICES

A. DATA FLOW DIAGRAM

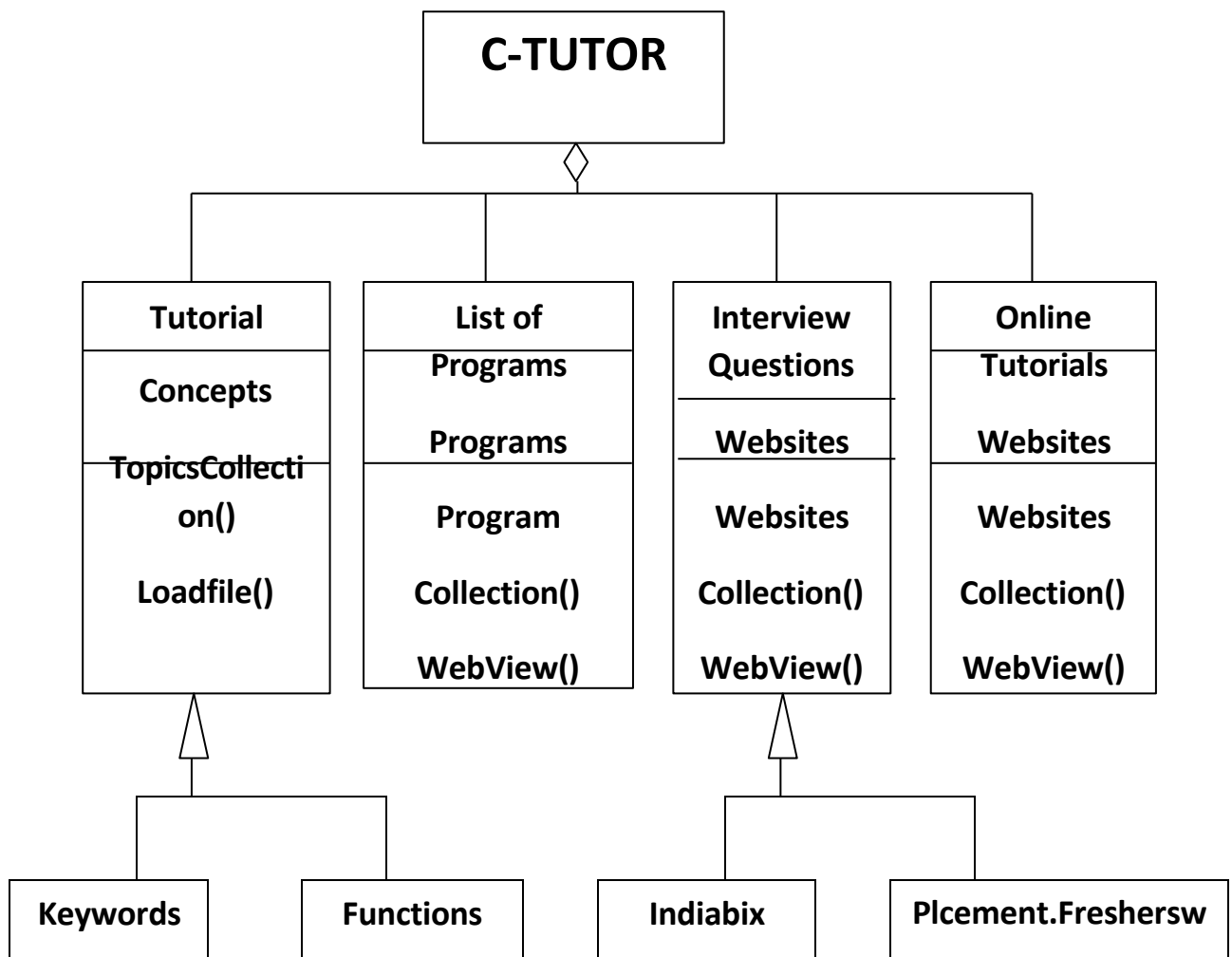
The following are the System architecture diagrams for an Android application. The following diagrams explain how an android application is created and how it can be installed in to an android mobile.



(Fig 1 DFD)

Class Diagram for C-Tutor

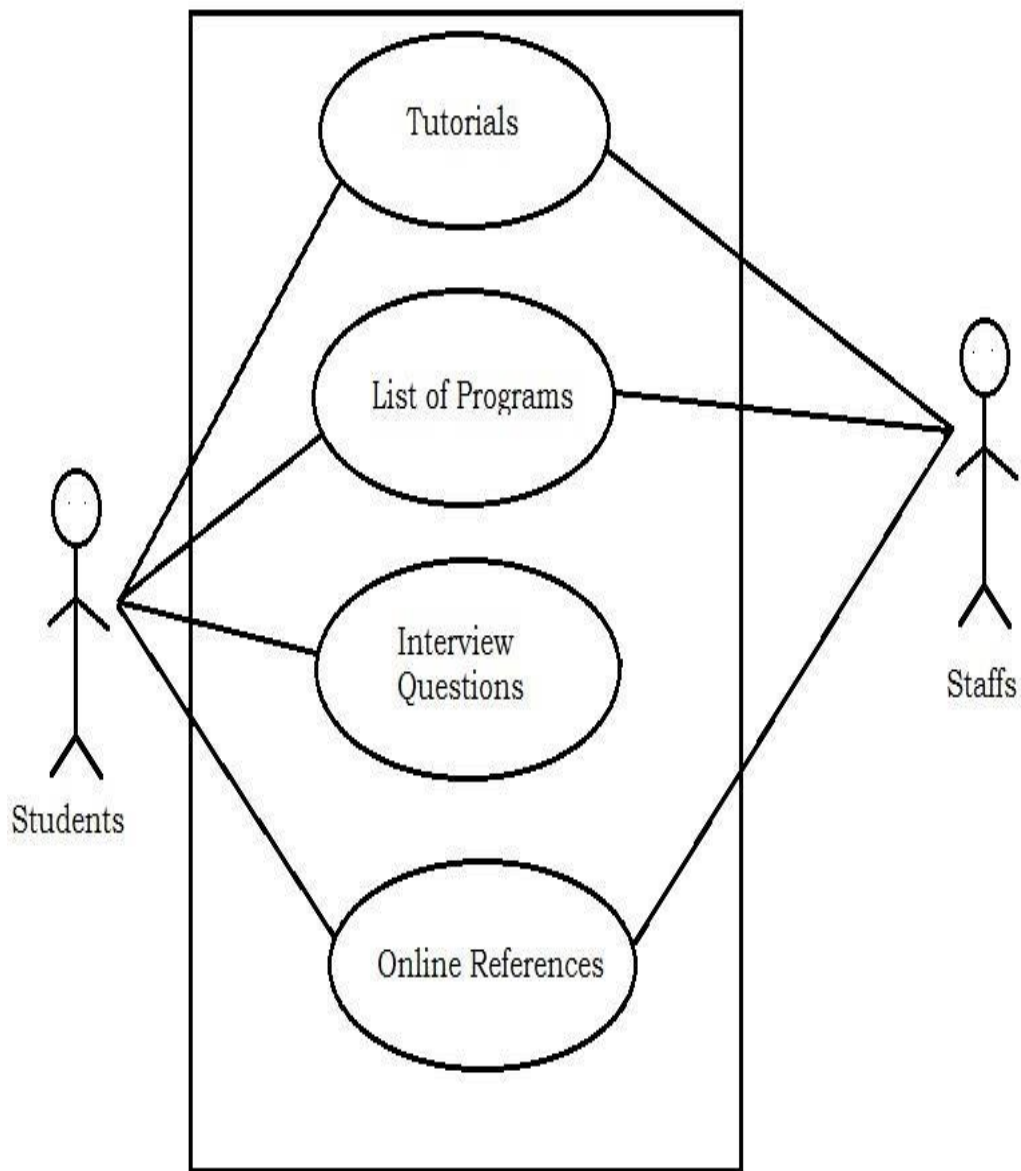
The following is the Class diagram for C-Tutor application that tells about how the modules are hierarchically connected.



(Fig: Class Diagram for C-Tutor)

Use Case Diagram for C-Tutor App:

The following is the use case diagram for C-Tutor application that contains Students and Staffs as actors and how they are accessing the use cases of our application.



B. TABLE STRUCTURE:-

	DVM	JVM
Architecture	Register	Stack
Os support	Android	Multiple
Executables	APK	JAR
Constant –Pool	Per Application	Per Class
Memory	Less	More

C.SAMPLE CODING

Main Activity code:

```
package com.example.clanguage;
import android.media.MediaPlayer;
import android.os.Bundle;
import android.os.CountDownTimer;
import android.app.Activity;
import android.content.Intent;
import android.view.Menu;
import android.view.View;

public class MainActivity extends Activity {
    public CountDownTimer countDownTimer;
    //public TextView text;
    private final long startTime = 3 * 1000;
    private final long interval = 1 * 1000;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        countDownTimer = new MyCountDownTimer(startTime, interval);

        countDownTimer.start();
        if (getIntent().getBooleanExtra("EXIT", false)) {
            finish();
        }
    }

    public void home(View v)
    {
        Intent in = new Intent(this, Home.class);
        startActivity(in);
    }
}
```

```

publicclass MyCountDownTimer extends CountDownTimer
{
    public MyCountDownTimer(long startTime, long interval)
    {
        super(startTime, interval);
    }

    publicvoid onFinish()
    {
        Intent in= new Intent(MainActivity.this,Home.class);
        startActivity(in);
    }
    @Override
    publicvoid onTick(long millisUntilFinished)
    {
        //text.setText("" + millisUntilFinished / 1000);
    }
}

```

Home Activity code:

```

package com.example.clanguage;
import android.os.Bundle;
import android.app.Activity;
import android.content.Intent;
import android.view.Menu;
import android.view.View;
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
import android.widget.Button;
publicclass Home extends Activity {
    @Override
    protectedvoid onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_home);
    }
}

```

```

final Animation animScale = AnimationUtils.loadAnimation(this, R.anim.scale);
Button b1 = (Button)findViewById(R.id.Button1);
Button b2 = (Button)findViewById(R.id.Button2);
Button b3 = (Button)findViewById(R.id.Button3);
Button b4 = (Button)findViewById(R.id.Button4);
final Intent in=new Intent(this,Tutorial.class);
final Intent in1=new Intent(this,Programs.class);
final Intent in2=new Intent(this,Interview.class);
final Intent in3=new Intent(this,Onlinetutorial.class);
b1.setOnClickListener(new Button.OnClickListener()
{
@Override
publicvoid onClick(View arg0) {
startActivity(in);
}});
b2.setOnClickListener(new Button.OnClickListener()
{
@Override
publicvoid onClick(View arg0) {
startActivity(in1);
}});
b4.setOnClickListener(new Button.OnClickListener()
{
@Override
publicvoid onClick(View arg0) {
startActivity(in3);
}});
b3.setOnClickListener(new Button.OnClickListener()
{
@Override
publicvoid onClick(View arg0) {
startActivity(in2);
}});
}}

```

Code for Loading File:

```
package com.example.clanguage;
import java.io.BufferedReader;
import java.io.InputStreamReader;
import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.webkit.WebView;
public class Introduction extends Activity {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_introduction);
        WebView helloTxt = (WebView)findViewById(R.id.webView1);
        String plainCode = readFileInAssetsDir("introduction.html");
        String htmlCode = "<pre>" + plainCode + "</pre>";
        helloTxt.loadDataWithBaseURL("", htmlCode, "text/html", "utf-8", "");
    }
    private String readFileInAssetsDir(String filename) {
        BufferedReader br = null;
        StringBuffer sb = new StringBuffer();
        try {
            br = new BufferedReader(new InputStreamReader(getAssets().open(filename)));
            String line;
            while((line = br.readLine()) != null)
                sb.append(line + "\n");
        } catch(Exception e) {
            // TODO
        }
        return sb.toString();
    }
}
```

Code for Loading Webpage:

```
package com.example.clanguage;
import android.net.Uri;
import android.os.Bundle;
import android.app.Activity;
import android.content.Intent;
import android.view.Menu;
import android.view.View;
import android.widget.Button;
public class Onlinetutorial extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_onlinetutorial);
        final Intent browse1 = new Intent( Intent.ACTION_VIEW , Uri.parse(
            "http://www.w3schools.in/c/intro/" ) );
        final Intent browse2 = new Intent( Intent.ACTION_VIEW , Uri.parse(
            "http://www.tutorialspoint.com/cprogramming/" ) );
        final Intent browse3 = new Intent( Intent.ACTION_VIEW , Uri.parse(
            "http://www.cprogramming.com/tutorial/c-tutorial.html" ) );
        final Intent browse4 = new Intent( Intent.ACTION_VIEW , Uri.parse(
            "http://www.programmingsimplified.com/c-program-examples" ) );
        final Intent browse5 = new Intent( Intent.ACTION_VIEW , Uri.parse(
            "http://fresh2refresh.com/cprogramming/" ) );
        final Intent browse6 = new Intent( Intent.ACTION_VIEW , Uri.parse(
            "http://www.programiz.com/c-programming" ) );

        Button b1 = (Button)findViewById(R.id.Button1);
        Button b2 = (Button)findViewById(R.id.Button2);
        Button b3 = (Button)findViewById(R.id.Button3);
        Button b4 = (Button)findViewById(R.id.Button4);
        Button b5 = (Button)findViewById(R.id.B5);
        Button b6 = (Button)findViewById(R.id.B6);
```

```

b1.setOnClickListener(new Button.OnClickListener()
{
@Override
publicvoid onClick(View arg0) {
startActivity(browse1);
}});
b2.setOnClickListener(new Button.OnClickListener()
{
@Override
publicvoid onClick(View arg0) {
startActivity(browse2);
}});
b3.setOnClickListener(new Button.OnClickListener()
{
@Override
publicvoid onClick(View arg0) {
startActivity(browse3);
}});
b4.setOnClickListener(new Button.OnClickListener()
{
@Override
publicvoid onClick(View arg0) {
startActivity(browse4);
}});
b5.setOnClickListener(new Button.OnClickListener()
{
@Override
publicvoid onClick(View arg0) {
startActivity(browse5);
}});
b6.setOnClickListener(new Button.OnClickListener()
{
@Override
publicvoid onClick(View arg0) {

```

```

        startActivity(browse6);
    });    }
}

```

XML code for Modules :

```

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="@color/MediumPurple"
    >
    <ScrollView
        android:id="@+id/scrollView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentRight="true"
        android:layout_alignParentTop="true">
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="535dp"
            android:background="@color/Khaki"
            android:orientation="vertical">
            <Button
                android:id="@+id/Button1"
                android:layout_width="match_parent"
                android:layout_height="125dp"
                android:layout_marginLeft="5dp"
                android:layout_marginRight="5dp"
                android:layout_marginTop="10dp"
                android:layout_weight="0.16"
                android:background="@drawable/b1"
                android:textSize="20dp"
                android:text="Tutorial"/>

```

<Button

```
android:id="@+id/Button2"  
android:layout_width="match_parent"  
android:layout_height="125dp"  
android:layout_marginLeft="5dp"  
android:layout_marginRight="5dp"  
android:layout_marginTop="5dp"  
android:background="@drawable/b1"  
android:layout_weight="0.16"  
android:textSize="20dp"  
android:text="List of Programs"
```

/>

<Button

```
android:id="@+id/Button3"  
android:layout_width="match_parent"  
android:background="@drawable/b1"  
android:layout_height="125dp"  
android:layout_marginLeft="5dp"  
android:layout_marginRight="5dp"  
android:layout_marginTop="5dp"  
android:layout_weight="0.16"  
android:textSize="20dp"  
android:text="Interview Questions"/>
```

<Button

```
android:id="@+id/Button4"  
android:background="@drawable/b1"  
android:layout_width="match_parent"  
android:layout_height="125dp"  
android:textSize="20dp"  
android:layout_marginLeft="5dp"  
android:layout_marginRight="5dp"  
android:layout_marginTop="5dp"  
android:layout_weight="0.16"  
android:layout_marginBottom="10dp"
```


android:text="Online Tutorials"/>

</LinearLayout>

</ScrollView>

</RelativeLayout>

XML code for Webview:

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout_width="fill_parent"

android:layout_height="fill_parent"

android:orientation="vertical">

<WebView

android:id="@+id/webView1"

android:layout_width="match_parent"

android:layout_height="330dp"

android:layout_weight="0.70"

android:textAlignment="center"/></LinearLayout>

D.SAMPLE INPUT AND OUTPUT

SCREEN LAYOUT:

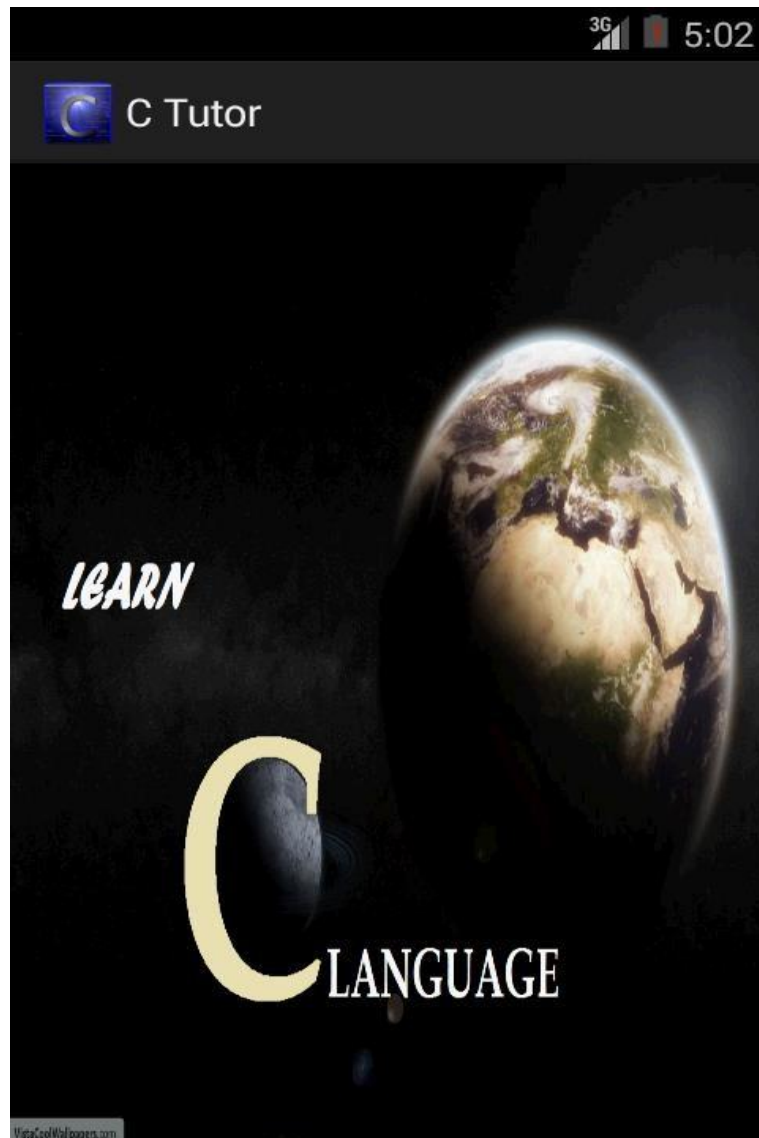


Fig:A.1 Home Screen

This is the home of our application which will automatically to our modules page within 3 seconds.

MODULE ACTIVITY:

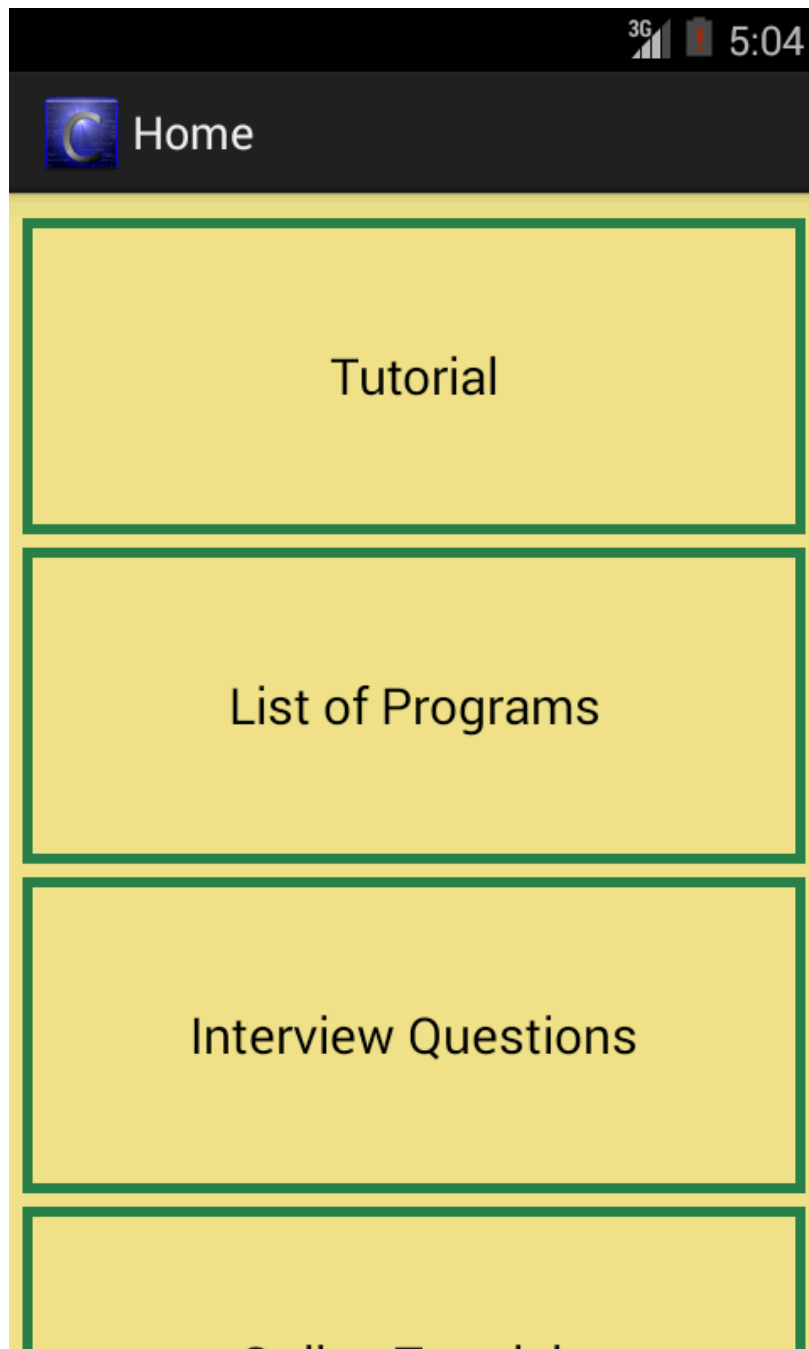


Fig A.2 Modules Screen

The above screen shot is for our application's Module Activity in which each module of our Application were provided in scroll view from which user can select a module he/she wants.

TUTORIALS ACTIVITY:

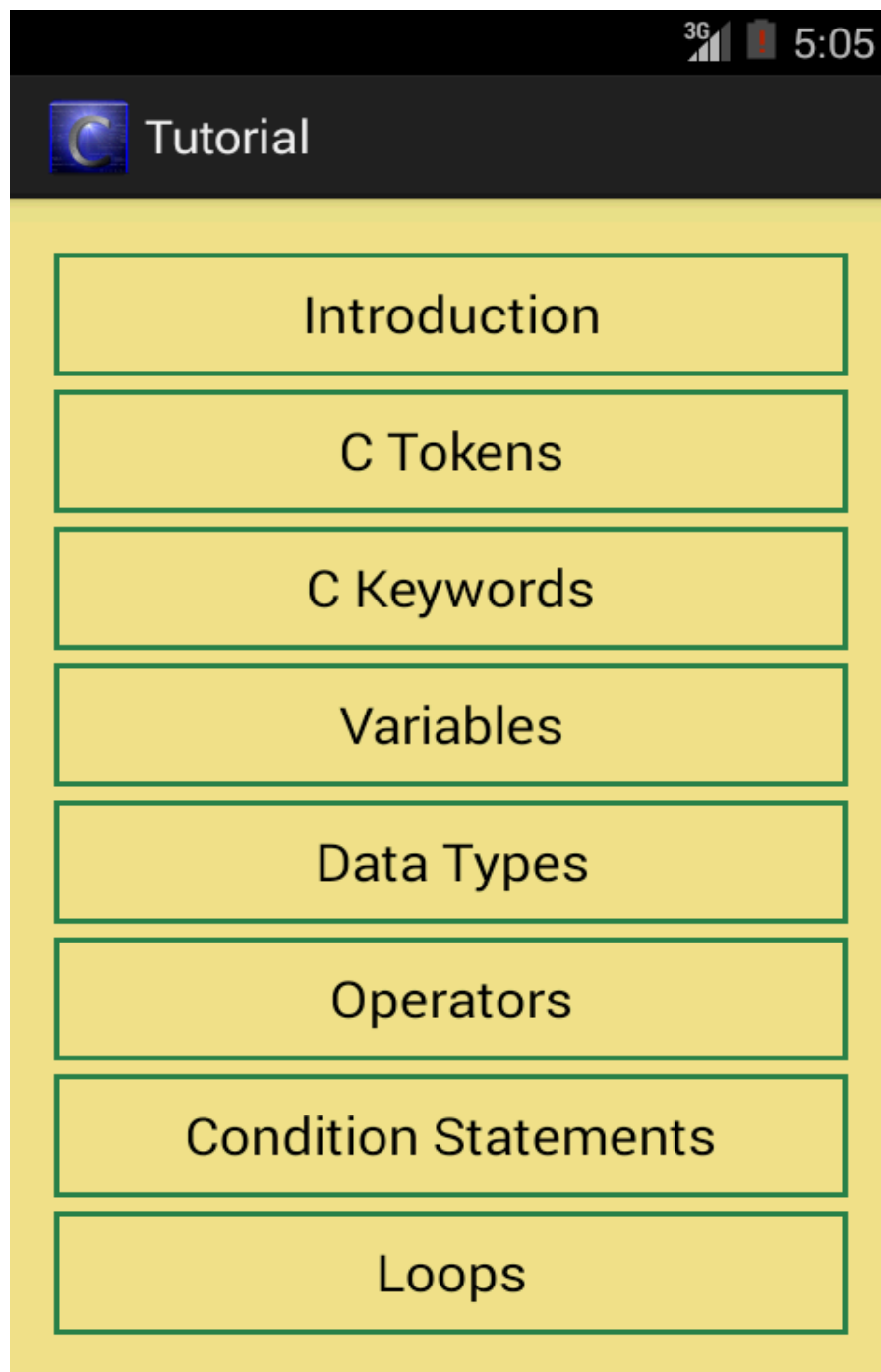


Fig A.3 Tutorials

The above screenshot is for Tutorials Activity, in which each concepts were provided in Scrollview from which user can select a concept he/she wants to read. The following are the screenshots for some concepts provided in our application,

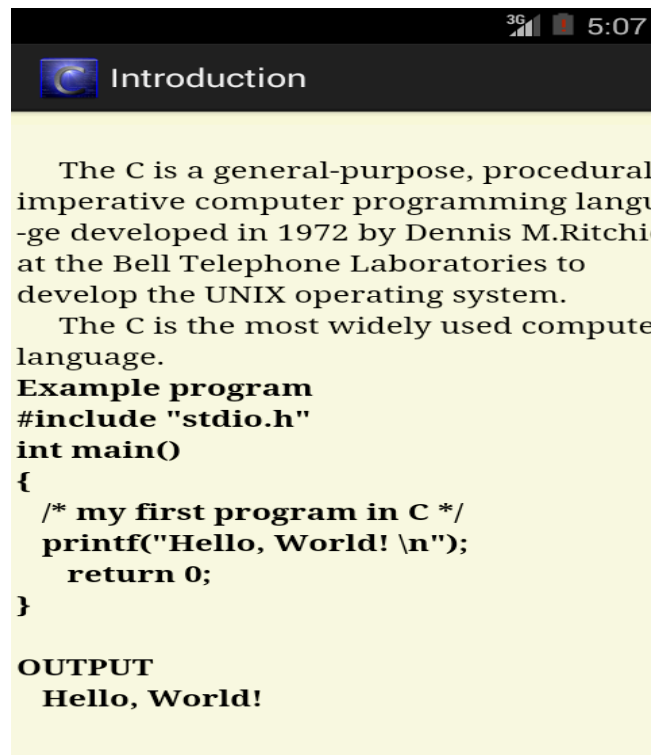
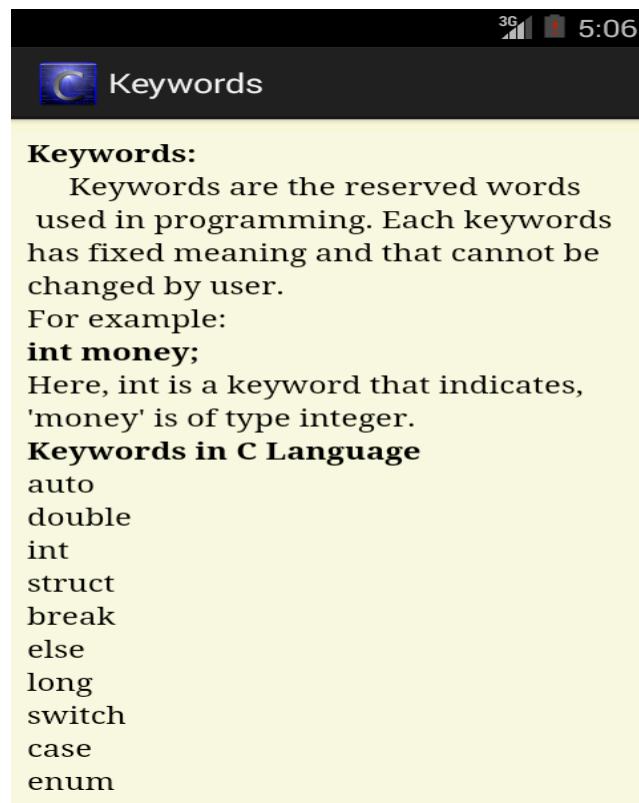


Fig A.4 Introduction



LIST OF PROGRAMS ACTIVITY:

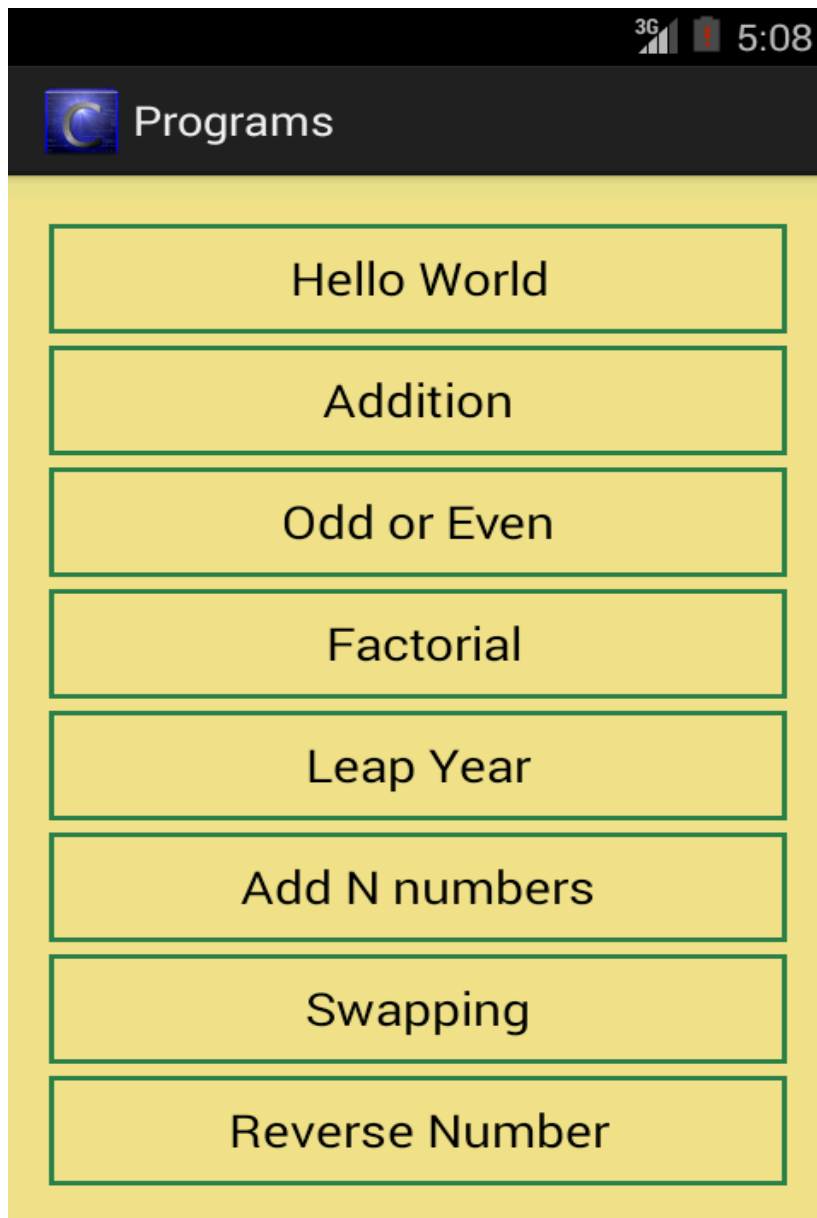



Fig A.6 List of Programs.

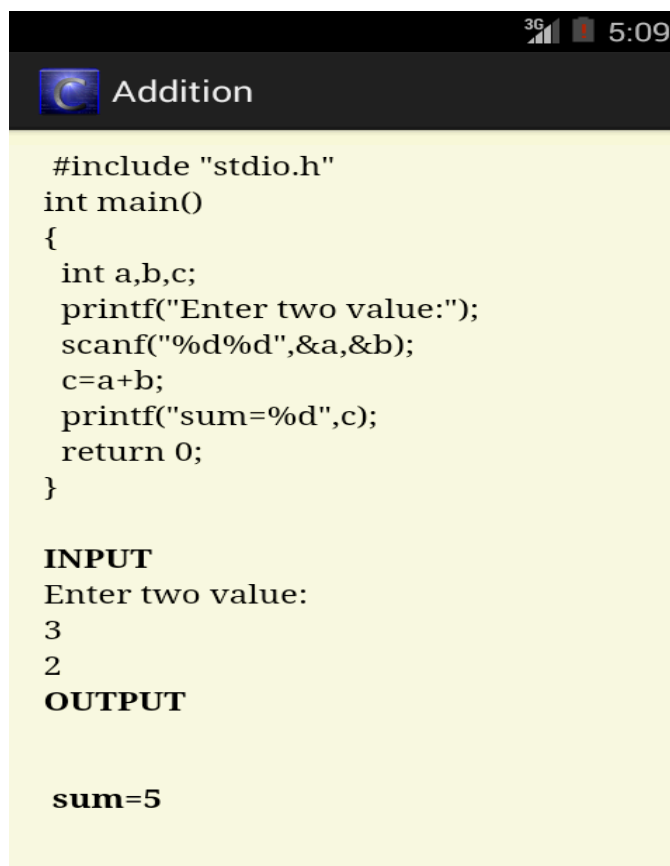


The screenshot shows a mobile application interface with a black header bar containing a blue 'C' logo and the text 'HelloWorld'. The main area has a yellow background and displays the following C code:

```
#include "stdio.h"
int main()
{
    printf("Hello World");
    return 0;
}
```

Below the code, the text **OUTPUT** is followed by 'Hello World'.

Fig A.7 Hello world



The screenshot shows a mobile application interface with a black header bar containing a blue 'C' logo and the text 'Addition'. The main area has a yellow background and displays the following C code:

```
#include "stdio.h"
int main()
{
    int a,b,c;
    printf("Enter two value:");
    scanf("%d%d",&a,&b);
    c=a+b;
    printf("sum=%d",c);
    return 0;
}
```

Below the code, the text **INPUT** is followed by 'Enter two value:', then the numbers '3' and '2' on separate lines. Below that, the text **OUTPUT** is followed by 'sum=5'.

Fig A.8 Addition Program

INTERVIEW QUESTIONS:

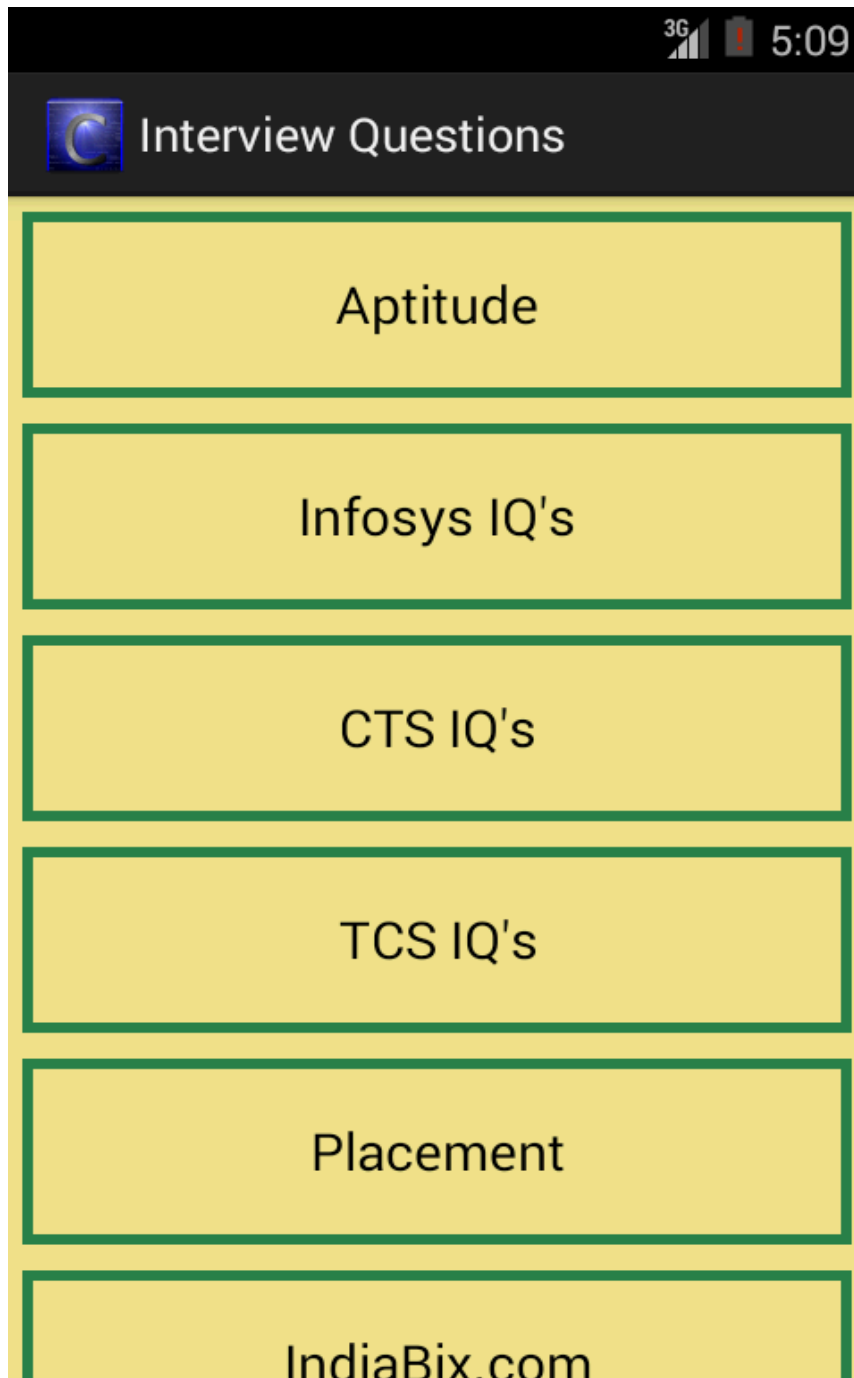


Fig A.9 Interview Questions

Above screenshot is for Interview questions activity from which user can choose a company name or website name to view the questions with answer for interview purpose.

ONLINE REFERENCES ACTIVITY:



Fig A.10 Online Tutorials

Above screenshot is for online tutorial activity from which user can choose a website name to view the online tutorials. The following are the screenshots for online tutorials provided in our application.

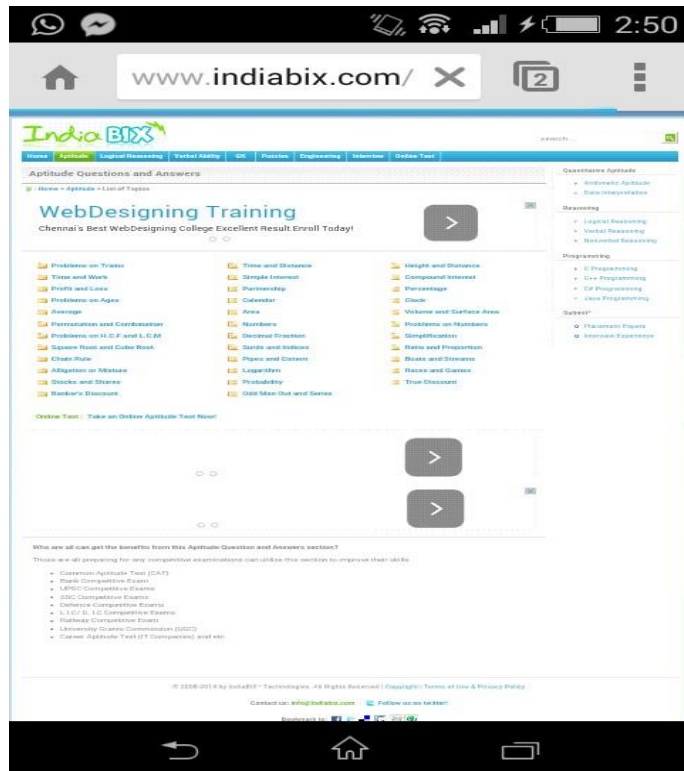


Fig A.11 Indiabix webpage



Fig A.11 W3Schools webpage