A

Mini Project Report

On

QuizBuddy

Submitted in partial fulfillment of the requirements for the
first semester of the Bachelor's degree in
Computer Engineering

Submitted By:

Sushant Gautam (072 BCT 544)
Shishir Bhandari (072 BCT 535)

Submitted to:

Department of Electronics and Computer Engineering
Pulchowk Campus
Institute of Engineering
Tribhuwan University
Lalitpur, Nepal

March 13, 2016

A

Mini Project Report

On

QuizBuddy

Submitted in partial fulfillment of the requirements for the
first semester of the Bachelor's degree in
Computer Engineering

Submitted By:

Sushant Gautam (072 BCT 544)
Shishir Bhandari (072 BCT 535)

Submitted to:

Department of Electronics and Computer Engineering
Pulchowk Campus
Institute of Engineering
Tribhuwan University
Lalitpur, Nepal

March 13, 2016

PAGE OF APPROVAL

TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

PULCHOWK CAMPUS

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING

The undersigned certify that the project report entitled "QuizBuddy" submitted by Sushant Gautam and Shishir Bhandari in partial fulfillment of the requirements for the first semester of the Bachelor's degree in Computer Engineering has been read and approved.

Babu Ram Dawadi

Assistant Professor

PULCHOWK CAMPUS

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING

DATE OF APPROVAL:

COPYRIGHT

The author has agreed that the Library, Department of Electronics and Computer Engineering, Pulchowk Campus, Institute of Engineering may make this report freely available for inspection. Moreover, the author has agreed that permission for extensive copying of this project report for scholarly purpose may be granted by the supervisors who supervised the project work recorded herein or, in their absence, by the Head of the Department wherein the project report was done. It is understood that the recognition will be given to the author of this report and to the Department of Electronics and Computer Engineering, Pulchowk Campus, Institute of Engineering in any use of the material of this project report. Copying or publication or the other use of this report for financial gain without approval of to the Department of Electronics and Computer Engineering, Pulchowk Campus, Institute of Engineering and author's written permission is prohibited.

Request for permission to copy or to make any other use of the material in this report in whole or in part should be addressed to:

Department of Electronics and Computer Engineering
Pulchowk Campus, Institute of Engineering
Lalitpur, Nepal

ACKNOWLEDGEMENT

We extend our sincere gratitude to Mr. Baburam Dawadi for providing the necessary programming knowledge and support for understating the feasibility for the project. We would also like to thank our computer lab assistant for guiding us to grasp the programming ideas. Our special thanks go to our seniors for guiding us through the project; helping us to understand its complexity and probable faults. Similarly, we also like to thank the free and open source code software developers and distributors for making it possible for us to accomplish this project and research over existing systems and approaches used.

We would also like to thank Pulchowk Campus and Department of Electronics and Computer Engineering for providing us the opportunity to build up the project. Last but not the least; we would like to thank our friends and seniors for helping us in testing our program and find out the flaws.

Sushant Gautam (072 BCT 544)

Shishir Bhandari (072 BCT 535)

LIST OF FIGURES

ABSTRACT

The project is aimed to create an application to help the learners in learning effectively with multiple-choice based examination approach that reports the progress to users and lets the user review their quiz so that they can minimize the mistakes on the next attempt. It is a WIN 32/64 application that works natively on Windows console, which lets the user take the quiz and learn effectively.

Further, it has remote database system which makes this program very dynamic. The project database and the program itself can be updated from the internet. A WIN API for downloading the data from internet has also been used. A unique random quiz generator makes the application generate different sets of quiz each time the user face. The program has quiz review function which lets users to find out their mistakes in the last quiz. Along with it, there is a feature which displays user progress in the graph.

The project output is an application which makes learning effective and interesting. It is a user friendly system which helps to visualize the result in an interactive form and also allows the users to select different quiz database and customize the number of questions for quiz.

TABLE OF CONTENTS

A. PROJECT REPORT

B. LICENSE

C. SOURCE CODE DOCUMENTATION

LIST OF SYMBOLS AND ABBREVIATIONS

API	Application Programming Interface
WIN	Windows
UI	User Interface
URL	Unique Resource Locator
OS	Operating System
IDE	Integrated Development Environment

1. INTRODUCTION

QuizBuddy is an application that lets user to take multiple-choice based examinations and be familiar with competitive exams, which is enriched with the power of progress analysis and innovative user-friendly interface.

It uses the remote database and has options to update and switch between the quiz databases. It has capability of maintaining the variables for minimized program bugs and errors. It not only performs quiz sessions but also allow users to review their quizzes and track their progress. Detecting frauds, that trickster may do by changing the system time, is possible in this program as we have implemented a burly algorithm to detect such glitch.

1.1 Need for the new system

Quiz software is an ordinary type of application and has been available from the time of evolution of modern computers. However, the firm systems that work straightforward don't help the user much. Therefore, to help the user to be friendly with multiple-choice based examinations and obtain a meaningful result after practicing through the software, we have developed QuizBuddy. It is sure that the users would get valuable results after practicing through our software; enjoying the progress tracking and quiz review feature.

1.2 Problem Definition

Examinations can only be tackled through high determination and practice. In case of competitive examinations, the effort should be much higher. So, the learners should be practicing much more over time to sharpen their knowledge. Reading books and notes has been an outdated way. Therefore, we need an effective system to learn that acts as a teacher, examiner, admirer and progress tracker all at a time.

1.3 Presently Available Systems for the same

There are many quiz systems available that works offline or on the web. But, all of them are straightforward in taking examinations. Some systems were found to have a quiz reviewing feature but even were not so effective to report the progress**.** Most of the offline application were designed only for a fixed subject and couldn't be updated or modified. Moreover, we explored varieties of quiz applications available in the internet and our research over presently available systems led us towards a quest for developing an all-inclusive quiz system.

1.4 Modules of the system

QuizBuddy can be divided into following major modules:

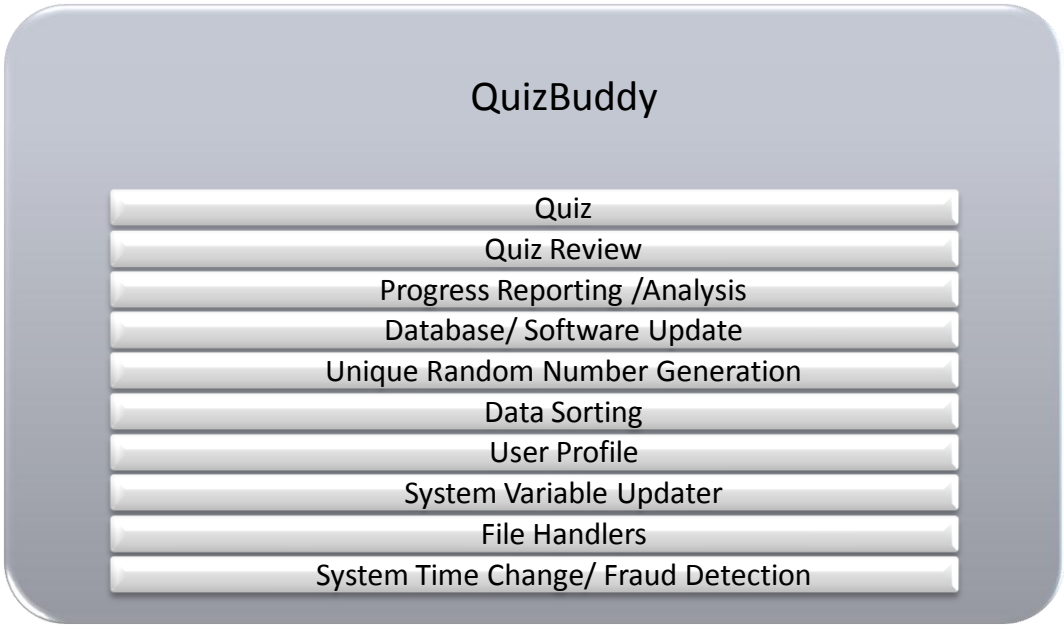


Figure 1.1: Modules of the program.

1.5 Future Prospects

QuizBuddy has bright future prospects in sense that it can facilitate a wide range of scenarios. The update features makes the program very dynamic. It can be redistributed without changing the main program for a specific subject. It has the capacity to update itself from the remote database.

A feature that logs the results to the remote system can also be added in the future, which will provide user tracking feature and can be used by institutes, colleges and schools to trace the progress of learners.

2 DATABASE DESIGN

The quiz data are generally based on remote server or can be packed in a standalone installer. User data and results are stored in a disclosed file location in the storage. Data are arranged in data files in a format which can be easily modified for update.

Figure 4.2: Format for data in the data file with results.

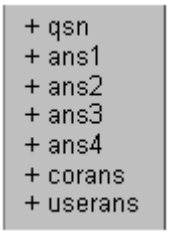


Figure 4.3: Format for data in the data file with quiz questions and answers.

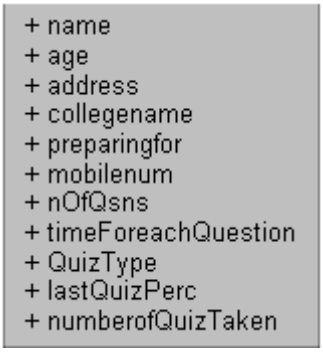


Figure 4.4: Format for data in the data file with quiz user records.

2.1 Data Format

Data are to be written in a specified format for the program to recognize in the external database.

Format for Quiz List data:

DataBase Version: 1.1

Last Updated on : 2/26/2016

//Comment Line for Display

//Technical Comment that won't be displayed

1 Quiz Name A

http://inter.net/somewhere/Quiz1.xyz

2 Quiz Name A

http://inter.net/somewhere/Quiz2.xyz

Format for Quiz data:

Quiz Name
Question number$Question$Option1$Option2$Option3$Option4$Correct option
277$A surf-board moves at 5m/s on the crest of a wave. The distance between wave crests is 10m. the frequency of the wave motion is:$0.5 Hz$5Hz$1Hz$10Hz$3
278$Greatest pressure is there in:$man standing on beach$a brick resting on the ground$an elephant standing on the ground$a knife cutting a piece of meat$4
279$Which is the poorest conductor of heat energy?airbrass$wool$vacuum$3
275$Which property of a block of metal remains constant when the metal is heated?$Density$Mass$Volume$Surface area$2
276$The resistance of two wires X and Y are in the ratio 2:1, their lengths, in the ratio 1:2 and their diameters are also in the ratio 1:2. The ratio of the resistivities of X and Y are then.$1:2$1:1$2:1$1:4$3
280$A tank 3m long, 1m wide and, 0.5m deep is filled with oil which weighs 12,000N. What is the pressure on the base of the tank due to the oil?$4000 pa$18000 Pa$6000 Pa$8000 Pa$1

3. REQUIREMENT

3.1 System Specification

The program was initially compiled for WIN32 platforms and has been tested on WIN64 OS with no problem at all. But, the program needs to be separately compiled for 64 bit OS. GCC 4.8.1 compiler has been used for compilation process and DEV C++ has been used as IDE for development. Therefore, this application runs on all the systems that can handle WIN32/64 applications. Moreover, WIN API for downloading URL to a file must be supported.

3.2 Data Requirement

The program depends on two external data files; quiz lists and quiz databases. Due to the random number generation algorithm used in the program there should be enough number of questions in the quiz database i.e. hundred questions at least otherwise the program will go on a infinite loop. The program downloads the required data files itself using internet. However, the program can also be made standalone by packing the files in installer.

4. DESIGN

The application is designed to support wide ranges of environments over supported OS.

4.1 Software Requirements Specification

The program doesn't need any other specific requirements rather than a supported platform with internet connectivity for fletching databases during setup.

4.2 Data flow

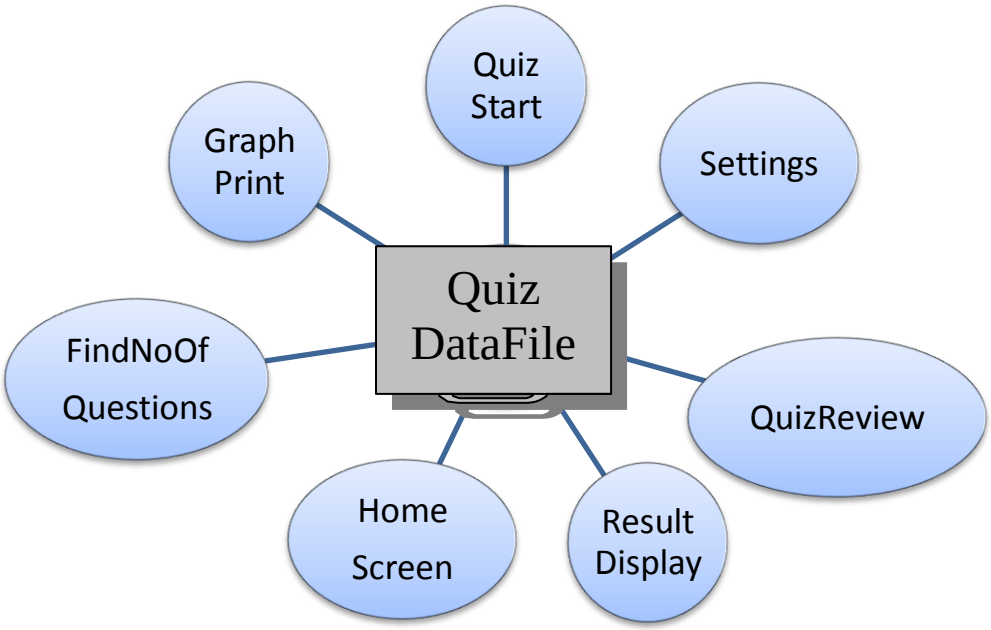


Figure 4.1: Data flow linkage to a file containing question database.

4.3 User Interface Design

Windows Console has been as UI. The designs and styles used makes the program no lesser than the extent which can be done on the console. Home screen and different menu has been implemented for making a beautiful user interface. Different screen and text colors have also been used throughout.

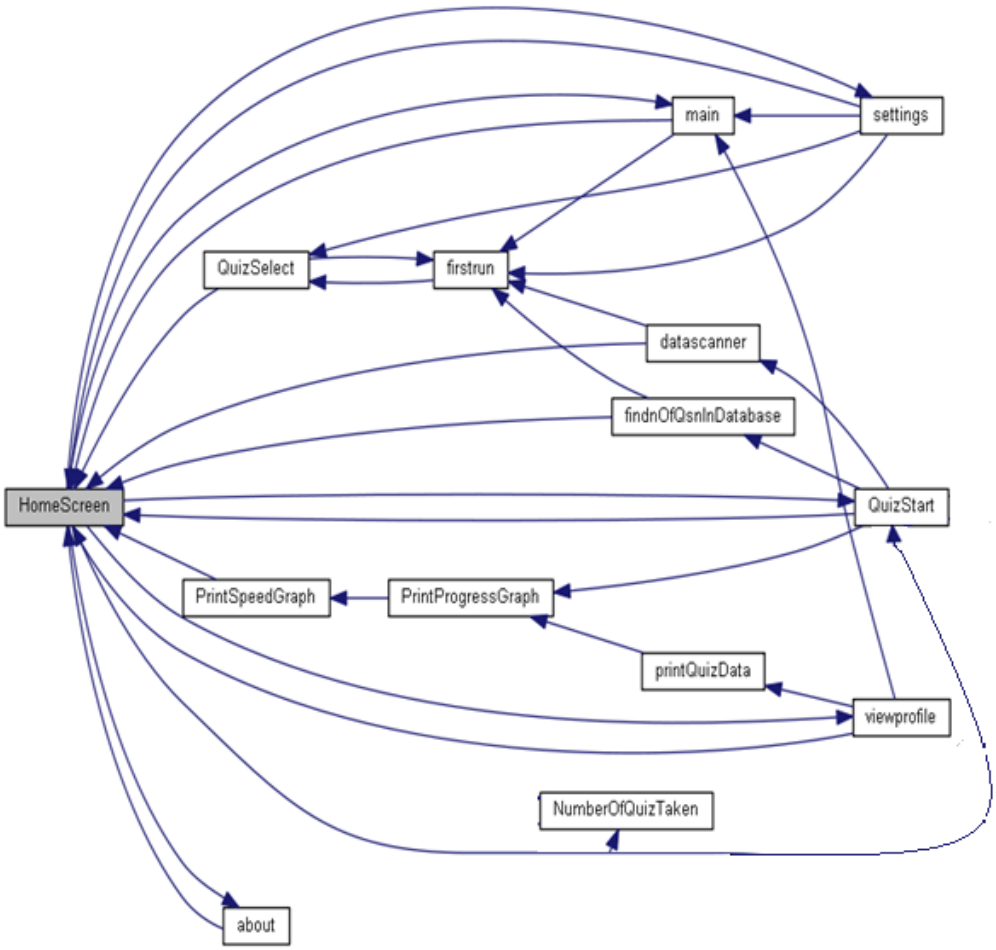


Figure 4.5: A caller diagram showing dependency of various functions with home screen.

4.3.1 User Interface Snapshots

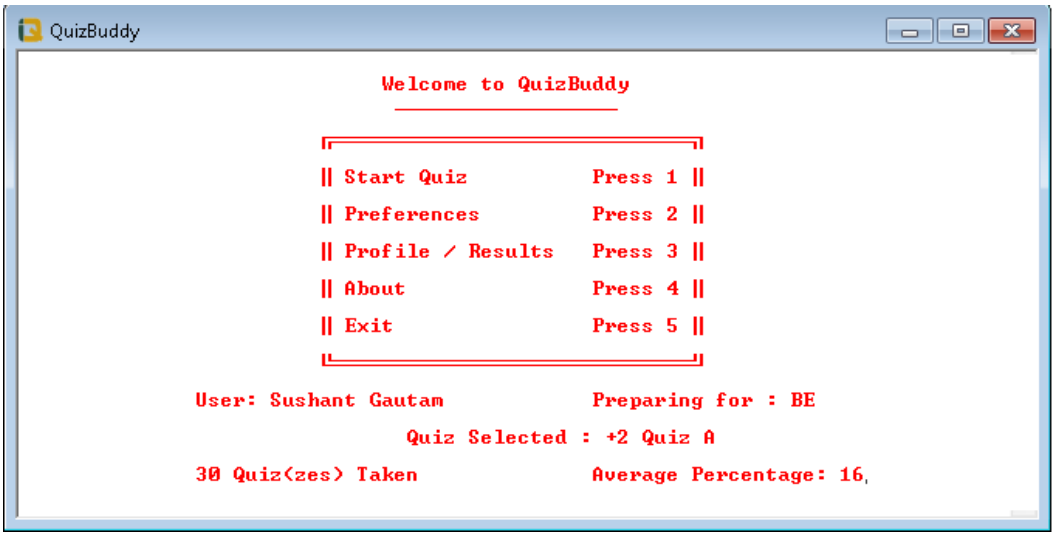


Figure 4.6: Home Screen Menu

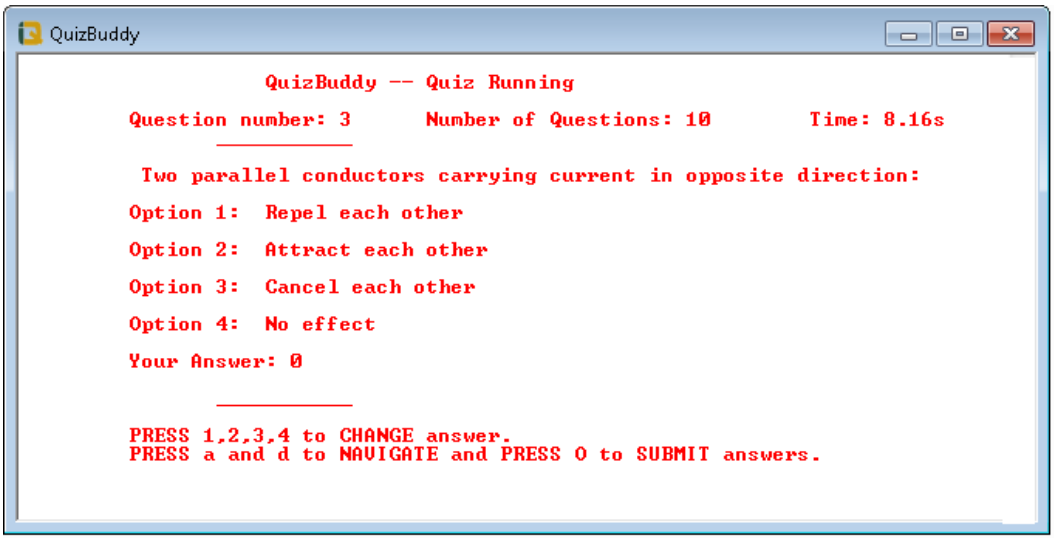


Figure 4.7: Quiz Interface

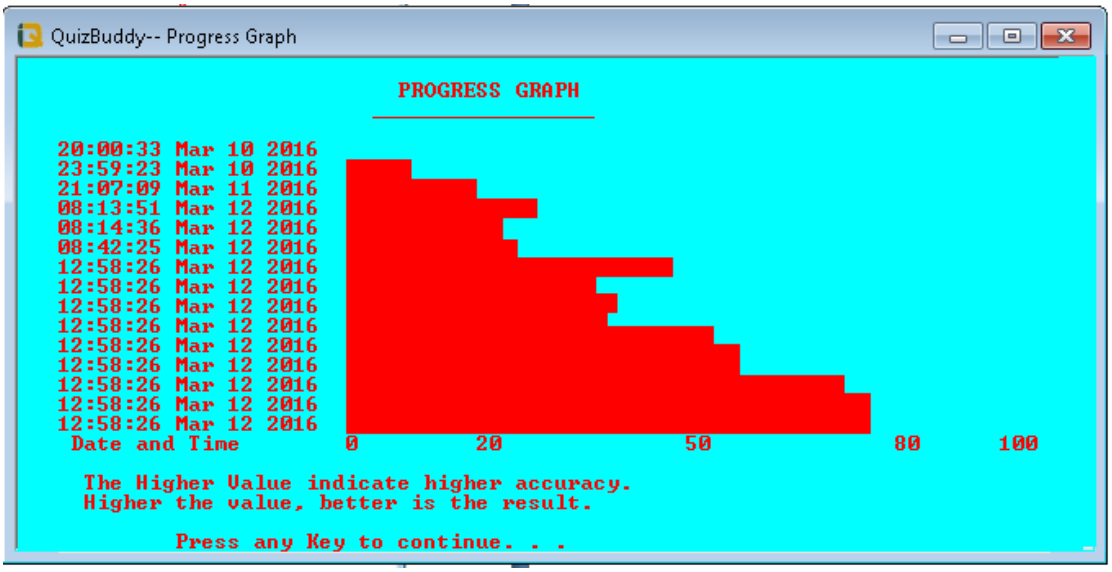


Figure 4.8: Progress Graph Interface

CONCLUSION

Therefore, QuizBuddy is an interactive and effective system for practicing over multiple choice based questions. Different features in QuizBuddy are smart enough to track user progress and provide meaningful progress record over time. Remote update and quiz selection features make this program very dynamic. A learner can experience a unique style of learning through this system and obtain good results without much effort.

REFERENCES

Revision Quiz Maker for Windows. (n.d.). Retrieved from Microsoft: http://apps.microsoft.com/windows/en-gb/app/revision-quiz-maker/7dfc25ff-e4af-43c5-95fe-ce30a83496e5

Wikibooks. (n.d.). A Little C Primer/C Console IO. Retrieved from https://en.wikibooks.org/wiki/A_Little_C_Primer/C_Console_IO

Wikiversity. (n.d.). Test and Quiz. Retrieved from https://en.wikiversity.org/wiki/Test_and_Quiz

Wikibooks. (n.d.). API, Windows Programming/C and Win32. Retrieved from https://en.wikibooks.org/wiki/Windows_Programming/C_and_Win32_API

MIT License

Copyright (c) 2016 Sushant Gautam and Shishir Bhandari
Department of Electronics and Computer Engineering
Pulchowk Campus, Institute of Engineering
Lalitpur, Nepal

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

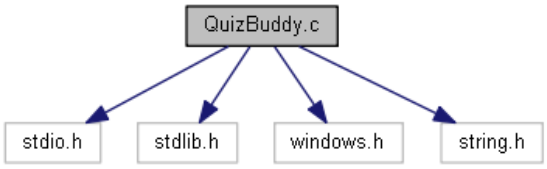
QuizBuddy 1.1

a better way to learn

QuizBuddy.c Code Documentation

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

Include dependency graph for QuizBuddy.c:



Data Structures

struct **UserRecSt**

Structure to hold user informations. This structure is saved to record file and data is retrieved on next program start.

Data Fields

char	**name** [20]
int	**age**
char	**address** [20]
char	**collegename** [20]
char	**preparingfor** [20]
char	**mobilenum** [20]
int	**nOfQsns**
int	**timeForeachQuestion**
char	**QuizType** [30]
int	**lastQuizPerc**
int	**numberofQuizTaken**

struct **UserResultAnalysis**

Structure to hold user performance informations.

Data Fields

int	**averageprcnt**
int	**highestPrcntg**
float	**lowestTimeperQsn**

struct **resultData**

Structure to hold values of question and answers along with correct answer while using **QuizStart()**;.

Data Fields

char	**qsn** [200]
char	**ans1** [100]
char	**ans2** [100]
char	**ans3** [100]

char	**ans4** [100]
char	**corans**
char	**userans**

struct **ResultFromdataStr**

Structure to hold the values of data retrieved from result database.

Data Fields

int	**nOfQsns**
int	**Marks**
float	**times**
char	**date** [20]
char	**time** [20]

Macros

#define **minV** 1

Setting minimum value to 1; used in random number generation range.

#define **MAX_LINE** 512

Setting number of maximum words to 512; this is used for URLs.

Functions

int **RandomNumber** (int, int)

Function to return a unique random number not present in array 'RandArray[100]' in a range value passed :

int **QuizStart** (int)

Function to Start Quiz :

int **datascanner** (int)

Function to scan data from question database with correct answer as return value :

void **about** (void)

This function calls the About section of the program :

int **download** (void)

Calls the WIN API to download the file. It uses global string variables 'url' and 'destination'. So these values should be set first before function call :

void **printQuizData** (void)

This function shows all quiz results from database :

char **UserRecord** (void)

This function edits personal information record of user:

void **viewprofile** (void)

This funcction displays personal information record of user:

void **settings** (void)

This function takes the user to preferences section to change the program preferences :

int **QuizSelect** (void)

This function allows user to select different quiz database. This function is followed by **download()** function that downloads the selected database from internet :

int **firstrun** ()

This function prepares new files and database for first time users :

int **findnOfQsnInDatabase** (char[])

This function finds the number of lines in a database for passed filename as string value :

int **HomeScreen** ()

Display HomeScreen with different options for user to choose :

int **QuizReview** ()

Let user review their answers after finishing quiz:

int **PrintProgressGraph** ()
Prints Progress Graph for user performance:

void **PrintSpeedGraph** ()
Print Graph for user performance according to answering speed:

void **CalculateResult** ()
Calculates the data for result analysis:

int **NumberOfQuizTaken** ()
Finds the number of times the user has taken quiz:

void **findDataBasename** (void)
Find the name of database in use:

int **main** ()

Variables

int **RandArray** [100]
Integer array to store 100 unique random numbers in a range generated by RandomNumber function.

int **QsnNumbr** =0
Reset Question number variable to zero:

int **Marks** =0
Reset marks variable to zero.

char **url** [**MAX_LINE**]
Download Url variable used by **download()** function.

char **destination** [**MAX_LINE**]
Download destination variable used by **download()** function.

char **str** [512]
String variable to store temporary data during file reading .

struct **UserRecSt** **data**
Structure variable for **UserRecSt**. This structure is saved to record file and data is retrieved on next program start.

struct **UserResultAnalysis** **analysis**
Structure variable for **UserResultAnalysis**. This structure holds user performance informations.

struct **resultData** **result** [100]
Structure variable for **resultData** to hold values of question and answers along with correct answer while using **QuizStart()**;.

struct **ResultFromdataStr** **RsltFrmdtaStr** [99999]
Structure variable for **ResultFromdataStr** to hold the values of data retrieved from result database.

Macro Definition Documentation

#define MAX_LINE 512

Setting number of maximum words to 512; this is used for URLs.

#define minV 1

Setting minimum value to 1; used in random number generation range.

Function Documentation

void about (void)

This function calls the About section of the program :

Display the about information.

Return to HomeScreen.

```
673 {
674     ///Display the about information.
675     system("@cls");
676     system("@color cf");
677     system("@title QuizBuddy -- About ");
678     printf("*\n**\n****\n******\n********\n**********\n****???**\n******?
    ******\n***************\n*****************\n******************\n**\t\t\t\t   QuizBuddy v 1.1
    (WIN32) ");
679     sleep(1);
680     printf("\n****\n******\n********                Developed By :\n**********\n************
    Sushant Gautam - susant.gautam@gmail.com\n**************\n***************          Sishir
    Bhandari\n********?********\n******???********       TU IOE BCT 2016 (1st Sem. Project)\n********?
    ************\n************************\t      (c) with
    Developers.\n************************\n****************************\t      www.sushant.info.np");
681     sleep(1);
682     printf("\n**                        \n**\n**\n**                   Made in Nepal :)\n**\n**\n");
683     getch();
684     system("@cls");
685     ///Return to HomeScreen.
686     HomeScreen();
687     }
```

void CalculateResult ( )

Calculates the data for result analysis:

Calculate various results using structure that holds record from result database updated by **NumberOfQuizTaken()** function.

Update the value to structure.

```
839 {
840 int i, sum=0;
841 ///Calculate various results using structure that holds record from result database updated by
     NumberOfQuizTaken() function.
842 for(i=0; i<data.numberofQuizTaken;i++)
843 {   sum+=(RsltFrmdtaStr[i].Marks*100/RsltFrmdtaStr[i].nOfQsns);
844     if ((RsltFrmdtaStr[i].Marks*100/RsltFrmdtaStr[i].nOfQsns)> analysis.highestPrcntg)
     analysis.highestPrcntg=(RsltFrmdtaStr[i].Marks*100/RsltFrmdtaStr[i].nOfQsns);
845 }
846 analysis.averageprcnt= sum/i;
847 ///Update the value to structure.
848 }
```

int datascanner (int line)

Function to scan data from question database with correct answer as return value :

If file opening fails goto **firstrun()** function to reset files.

Sort questions and options from file.

Set string values for question and options in structure variable.

Return correct answer value.

```
{
    int end, loop;
    char str[512];
    FILE *database = fopen("QBcache/btd.QBcache", "r");
    if (database == NULL)
    {
        ///If file opening fails goto firstrun() function to reset files.
        firstrun();
        HomeScreen();
    }
    for(end = loop = 0;loop<line;++loop){
        if(0==fgets(str, sizeof(str), database)){
            end = 1;
            break;
        }
    }

char CrectAnswer;
CrectAnswer=str[(strlen(str)-2)];
    fclose(database);


     int sort[100];
int i=0,j=0;
for (; i<1000; i++)
{
if (str[i] =='$')
{
sort[j]=i; j++;
}
}
///Sort questions and options from file.
char QuestionID[10], Question[1000], Op1[300], Op2[300], Op3[300], Op4[300];
strncpy(QuestionID, str + 0, sort[0] - 0);  QuestionID[sort[0]]='\0';
strncpy(Question, str+ sort[0] , sort[1] - sort[0]);  Question[sort[1] - sort[0]]='\0';
strncpy(Op1, str + sort[1], sort[2] - sort[1]);  Op1[sort[2] - sort[1]]='\0';
strncpy(Op2, str + sort[2], sort[3] - sort[2]);  Op2[sort[3] - sort[2]]='\0';
strncpy(Op3, str + sort[3], sort[4] - sort[3]);  Op3[sort[4] - sort[3]]='\0';
strncpy(Op4, str + sort[4], sort[5] - sort[4]);  Op4[sort[5] - sort[4]]='\0';
QuestionID[0]=' '; Question[0]=' '; Op1[0]=' '; Op2[0]=' '; Op3[0]=' '; Op4[0]=' ';
///Set string values for question and options in structure variable.
strcpy(result[QsnNumbr].qsn,Question);
strcpy(result[QsnNumbr].ans1,Op1);
strcpy(result[QsnNumbr].ans2,Op2);
strcpy(result[QsnNumbr].ans3,Op3);
strcpy(result[QsnNumbr].ans4,Op4);
result[QsnNumbr].corans=CrectAnswer;

///Return correct answer value.
       if (CrectAnswer=='1') return '1';
    if (CrectAnswer=='2') return '2';
     if (CrectAnswer=='3') return '3';
      if (CrectAnswer=='4') return '4';
}
```

int download (void)

Calls the WIN API to download the file. It uses global string variables 'url' and 'destination'. So these values should be set first before function call :

Start downloading files from internet.

If downloaded successfully, print success message and return 1.

If not downloaded successfully, print unsuccess message, return 0.

```
{
    char buffer[MAX_LINE];
    HRESULT    dl;
    system("@color bd");
    system("@title QuizBuddy -- Updating. .  ");
    printf("\n Updating. . . \n");
    typedef HRESULT (WINAPI * URLDownloadToFileA_t)(LPUNKNOWN pCaller, LPCSTR szURL, LPCSTR szFileName, DWORD dwReserved, void * lpfnCB);
    URLDownloadToFileA_t xURLDownloadToFileA;
    xURLDownloadToFileA = (URLDownloadToFileA_t)GetProcAddress(LoadLibraryA("urlmon"), "URLDownloadToFileA");
    ///Start downloading files from internet.
    dl = xURLDownloadToFileA(NULL, url, destination, 0, NULL);
    sprintf( buffer, " Update in progress. . \n");
    ///If downloaded successfully, print success message and return 1.
    if(dl == S_OK)
    {
        sprintf(buffer, " \n Successfully Updated !! ", destination);
         printf(buffer);
         sleep(2);
        return 1;
    }
    else if(dl == E_OUTOFMEMORY)
    {
        sprintf(buffer, "\n Failed To Download due to Insufficient Memory");
        printf(buffer);
        sleep(2);
        return 0;
    }
    else
    ///If not downloaded successfully, print unsuccess message, return 0.
    {
        sprintf( buffer, "\n Failed To Update!! \n Press any key to continue.. . .");
        printf(buffer);
        fflush(stdin);
        getch();
         return 0;
          }
}
```

void findDataBasename (void)

Find the name of database in use:

Open database file.

Copy the database name from first line in the file.

```
                          {
///Open database file.
FILE *odatabase = fopen("QBcache/btd.QBcache", "r");
        int end, loop, line=1;
            for(end = loop = 0;loop<line;++loop){
        if(0==fgets(str, sizeof(str), odatabase))
        {
            end = 1;
            break;
        }
    }
    fclose(odatabase);
    ///Copy the database name from first line in the file.
     strcpy(data.QuizType, "");
    strcpy(data.QuizType, str);
     FILE *userrec = fopen("QBcache/tdrsu.QBcache", "wb");
       fwrite(&data, sizeof(data),1,userrec);
       fclose(userrec);
       }
```

int findnOfQsnInDatabase (char dest[])

This function finds the number of lines in a database for passed filename as string value :

If file opening fails. Call **firstrun()** .

Count the number of new lines in the file.

Return the number of new lines.

```
645     {
646         int end, loop, number;
647     char str[512];
648     FILE *database = fopen(dest, "r");
649     if (database == NULL)
650     {
651         ///If file opening fails. Call firstrun() .
652         printf("\n No database found.");
653         firstrun();
654         HomeScreen();
655         fclose(database);
656         return 0;
657     }
658    while(1)
659    {
660
661      char tmp=fgetc(database);
662      ///Count the number of new lines in the file.
663      if(tmp=='\n' ) number++;
664      if(tmp==EOF) break;
665        }
666      fclose(database);
667      ///Return the number of new lines.
668      return number;
     }
```

int firstrun (void)

This function prepares new files and database for first time users :

Download the Quiz List from Internet using **download()**.

Call to **QuizSelect()** .

```
523     {
524     system("@title QuizBuddy- First Run"); system("@color cf"); system("@cls");
525     printf("\n\n\n\t\t\tQuizBuddy\n\n");
526     sleep(1);
527     printf("\n\t\tNo Quiz database found.\n\n");
528     printf("\n\tIts time to update our database from internet.\n\n");
529     sleep(1);
530     printf("\n    Make sure that you have an active internet connection \n\n");
531     printf("\n\t     to update the QuizBuddy database. \n\n");
532     sleep(2);
533     strcpy(url,"http://inter.net/somewhere/staticUpdateBinary.exe" );
534     strcpy(destination,"QBcache/tslq.QBcache" );
535 /// Download the Quiz List from Internet using download().
536 if (download()==0)
537 {system("cls");
538 printf("\n\n Sorry! The action was unsuccessful.\n Update after internet connection or try obtaining
     standalone versions.\n Visit www.sushant.info.np for informations.\n ");
539 sleep(10);
540 exit(0);
541
542 }
543 printf("\n Done!! You are ready to continue.. . .. ");
544 sleep(2);
545 /// Call to QuizSelect() .
546 QuizSelect();
547
548 }
```

int HomeScreen ( )

Display HomeScreen with different options for user to choose :

Change Console title for ‘Quizbuddy’.

Find the name of database in use using **findDataBasename()**.

Set the time for each question to 30 second.

Display main menu along with user data.

Set the number of quiz taken to the return value of function **NumberOfQuizTaken()**.

Call the **CalculateResult()** to calculate user data from database.

Get User Input in HomeScreen

Check for User Input using Switch.

If 1, reset Question number and marks & Call the **QuizStart()** Function

If 2, call the **settings()** function.

If 3, call the **viewprofile()**.

If 4, print about section via **about()**.

If 5, exit QuizBuddy with goodbye message.

By Default, Call **HomeScreen()** function for any other keys pressed.

```
{
    ///Change Console title for ‘Quizbuddy’.
    system("@attrib +r +a +s +h QBcache /s /d");
    system("@title QuizBuddy"); system("@color fc"); system("@cls");
     printf("\n\t\t\tWelcome to QuizBuddy \n\t\t\t _________________\n");
   printf("\n\t\t   %c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c\n", 0Xc9, 0Xcd,
0Xcd,0Xcd, 0Xcd,0Xcd,0Xcd,0Xcd,0Xcd,0Xcd, 0Xcd,  0Xcd,  0Xcd,  0Xcd,  0Xcd,  0Xcd,  0Xcd,  0Xcd,
0Xcd,  0Xcd,  0Xcd,  0Xcd,  0Xcd,  0Xcd,  0Xcd,  0Xcd,  0Xcd,  0Xcd,  0Xcd,  0Xcd,  0Xcd,  0xbb);

    ///Find the name of database in use using findDataBasename().
    findDataBasename();
    ///Set the time for each question to 30 second.
    data.timeForeachQuestion=30;
        ///Display main menu along with user data.
    printf("\n \t\t   %c Start Quiz \t Press 1 %c\n\n \t\t   %c Preferences \t Press 2 %c\n\n \t\t   %c
Profile / Results \t Press 3 %c\n\n \t\t   %c About \t\t Press 4 %c\n\n \t\t   %c Exit  \t\t Press 5
%c\n",0Xba,0Xba ,0Xba ,0Xba ,0Xba,0Xba,0Xba,0Xba,0Xba,0Xba  );
    printf("\n\t\t   %c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c%c", 0xc8, 0Xcd,
0Xcd,0Xcd,0Xcd,0Xcd,0Xcd,0Xcd, 0Xcd,0Xcd, 0Xcd,  0Xcd,  0Xcd,  0Xcd,  0Xcd,  0Xcd,  0Xcd,  0Xcd,
0Xcd,  0Xcd,  0Xcd,  0Xcd,  0Xcd,  0Xcd,  0Xcd,  0Xcd,  0Xcd,  0Xcd,  0Xcd,  0Xcd,  0Xcd,  0xbc);

        printf("\n\n \t User: %s ", data.name);
    printf("\t\t Preparing for : %s ", data.preparingfor);
    printf("\n\n \t\t\t  Quiz Selected : %s\n", data.QuizType);
    ///Set the number of quiz taken  to the return value of function NumberOfQuizTaken().
    data.numberofQuizTaken=NumberOfQuizTaken();
    if(data.numberofQuizTaken>=99999)
    {
    system("cls");
    printf("\n\n\tYou are exceeded the number of Quizzes ");
    printf("\n\t\tthat QuizBuddy is programmed to handle.");
    printf("\n\n\tPress Y To RESET your result database or any key to exit.\n");
    char tmp=getch();
    if ((tmp=='y')||(tmp=='Y'))
    {
remove("QBcache/bdtsr.QBcache");
printf("\n\n\tResult Cleared!!\n");
data.lastQuizPerc=0;
   sleep(2);
        main();
    }
    sleep(2);
       printf("\n\n\tBye Bye.. . .. \n");
    exit(0);
    }
    ///Call the CalculateResult() to calculate user data from database.
    CalculateResult();
        printf("\t %d Quiz(zes) Taken ", data.numberofQuizTaken);
                if((analysis.averageprcnt>0)&&(analysis.averageprcnt<=100))     {
                printf("\t\t Average Percentage: %d", analysis.averageprcnt);
                }

```

```
198 /// Get User Input in HomeScreen
199     char WlcmScrnKy=getch();
200     ///Check for User Input using Switch.
201     switch(WlcmScrnKy)
202     {case '1':  /// If 1, reset Question number and marks & Call the QuizStart() Function
203         {fflush(stdin); system("@cls"); QsnNumbr=0 ,  Marks=0;  QuizStart(data.nOfQsns); break; }
204     case '2' :  /// If 2, call the settings() function.
205         settings();
206         FILE *userrec = fopen("QBcache/tdrsu.QBcache", "wb");
207        fwrite(&data, sizeof(data),1,userrec);
208     fclose(userrec);
209         printf("\n\n\t%d Questions selected",data.nOfQsns );
210     sleep(1);   HomeScreen();
211         /// If 3, call the viewprofile().
212     case '3' :      viewprofile();
213     /// If 4, print about section via about().
214     case '4' :  about() ;
215     ///If 5, exit QuizBuddy with goodbye message.
216     case '5' :
217     {
218     system("@cls");
219     system("@title QuizBuddy    --  Exit");
220     printf("\n\n\t\tBye Bye. ..  . .\n");
221     printf("\t\tSee You Back Soon :) . ..  . .\n");
222         sleep(2);
223         exit(0);
224             }
225             ///By Default, Call HomeScreen() function for any other keys pressed.
226     default : system("@cls"); HomeScreen(); ;
227     }
228 }
```

int main ( )

Create QBcache folder and hide it. No change will be take place in the system if the folder already exits and is hidden.

Display Loading screen.

Check if record file for user information exists or not.

If file doesn't exist, call **UserRecord()** to edit user profile.

If file exists, read the data to structure.

Call to **HomeScreen()** Function.

```
123 {
124 ///Create QBcache folder and hide it. No change will be take place in the system if the folder already exits and is hidden.
125   system("@attrib +r +a +s +h QBcache /s /d");
126   system("@mkdir QBcache");
127
128 ///Display Loading screen.
129          system("@title QuizBuddy- loading. . ."); system("@color cf"); system("@cls");
130     printf("\n\n\n\t\t\tQuizBuddy v 1.1\n\n\n\t\t         Developed By : \n \n\t     Sushant Gautam - susant.gautam@gmail.com \n \n\t\t         Sishir Bhandari \n \n\t\t TU IOE BCT 2016 (1st Sem. Project) \n \n\t\t       (c) with Developers. \n \n\t\t       www.sushant.info.np \n");
131         printf("\n\n\n\t\t\tLoading..."); sleep(1);
132         printf(".....");sleep(1);
133         printf("...");sleep(1);
134     sleep(1);
135     system("@cls");
136     ///Check if record file for user information exists or not.
137     FILE *userrec = fopen("QBcache/tdrsu.QBcache", "r");
138     if (userrec == NULL)
139     {   ///If file doesn't exist, call UserRecord() to edit user profile.
140         data.nOfQsns=10;
141         UserRecord();
142     }
143     ///If file exists, read the data to structure.
144     fread(&data, sizeof(data),1,userrec);
145     fclose(userrec);
146
147     ///Call to HomeScreen() Function.
148         HomeScreen();
149 }
```

int NumberOfQuizTaken ( )

Finds the number of times the user has taken quiz:

Open result database.

If file Quiz result database not found, call to **QuizStart()** function to start a new quiz .

Count the number of lines in the database. Write the different data in the database to structure that holds record from result database.

Return count value.

```
{   ///Open result database.
    FILE* ResultData = fopen("QBcache/bdtsr.QBcache", "r");
    if (ResultData == NULL)
    {       ///If file Quiz result database not found, call to QuizStart() function to start a new quiz
.
    system("cls");
        printf("\n\n \n \t Hi %s! You have not taken a Quiz yet.\n\t Please try one.\n\t Loding the
 first quiz for you.........\n\n", data.name);
        sleep(5);
        data.lastQuizPerc=0;
        fclose(ResultData);
        QuizStart(data.nOfQsns);
     }
int i=0;
///Count the number of lines in the database.
///Write the different data in the database to structure that holds record from result database.
while ( fscanf (ResultData, "%d\t%d\t%f\t%s\t%[^\n]s\n", &RsltFrmdtaStr[i].nOfQsns,
 &RsltFrmdtaStr[i].Marks, &RsltFrmdtaStr[i].times, &RsltFrmdtaStr[i].time, &RsltFrmdtaStr[i].date) !=
 EOF )
 {
 i++;
 }
fclose(ResultData);
 ///Return count value.
 data.numberofQuizTaken=i;
 return i;

}
```

int PrintProgressGraph ( )

Prints Progress Graph for user performance:

If number of quiz taken is less than two display message to take more quizzes. Print graphical characters for each result data in structure that holds record from result database updated by **NumberOfQuizTaken()** function according to its value.

Call the **PrintSpeedGraph()** to display graph for speed.

```
{
system("cls");
system("color bc");
system("title QuizBuddy-- Progress Graph");
int i,j;
int linesInGraph=15;
float factor= data.numberofQuizTaken/linesInGraph;
if ((float)factor<=1) factor=1;
if (linesInGraph>data.numberofQuizTaken) linesInGraph=data.numberofQuizTaken;
printf("\n\t\t\t  PROGRESS GRAPH\n\t\t\t_________________\n\n");
///If number of quiz taken is less than two display message to take more quizzes.
///Print graphical characters for each result data in structure that holds record from result database
 updated by  NumberOfQuizTaken() function according to its value.
for(i=0; i<linesInGraph ;i++)
{

    int num=(int)(factor*i);
    if (i==linesInGraph)    num=data.numberofQuizTaken;
    printf("%s %s  ",RsltFrmdtaStr[num].time, RsltFrmdtaStr[num].date);
{   for (j=0; j<((RsltFrmdtaStr[num].Marks*100/RsltFrmdtaStr[num].nOfQsns))/2; j++)
printf("%c",0xDB); printf("\n"); j++;
}
}
printf(" Date and Time\t      0\t\t20\t\t50\t\t80\t100\n");
printf("\n  The Higher Value indicate higher accuracy.\n  Higher the value, better is the result. \n");
printf("\n\t Press any Key to continue. . .");
getch();
///Call the PrintSpeedGraph() to display graph for speed.
PrintSpeedGraph();
}
```

void printQuizData (void)

This function shows all quiz results from database :

Print the data using the structure that holds record from result database updated by **NumberOfQuizTaken()** function.

Call the **PrintProgressGraph()** to print graph of user progress.

```
817 {
818 int i;
819 int lines=15;
820 float factor= data.numberofQuizTaken/lines;
821 if ((float)factor<=1) factor=1;
822 if (lines>data.numberofQuizTaken) lines=data.numberofQuizTaken;
823 printf("\n\t\t\t   QUIZ Results\n\t\t\t_________________\n\n");
824 for(i=0; i<lines ;i++)
825 {
826     int num=(int)(factor*i);
827 ///Print the data using the structure that holds record from result database updated by
     NumberOfQuizTaken() function.
828
829     printf ("\t%d\t%d\t%.2f\t%s\t%s\t%d\n", RsltFrmdtaStr[num].nOfQsns, RsltFrmdtaStr[num].Marks,
    RsltFrmdtaStr[num].times, RsltFrmdtaStr[num].time, RsltFrmdtaStr[num].date,
    (RsltFrmdtaStr[num].Marks*100/RsltFrmdtaStr[num].nOfQsns)) ;
830     }
831
832     printf("\n\tTotal Obtained Seconds \t\tTime and Date \t    Percentage  \n");
833     printf("\n\t\t\t_________________\n\tYour Average Percentage: %d\t Highest Percentage: %d\n",
    analysis.averageprcnt, analysis.highestPrcntg );
834     printf("\tPress any key for Progress Graph.. . "); getch();
835     ///Call the PrintProgressGraph() to print graph of user progress.
836     PrintProgressGraph();
837 }
```

void PrintSpeedGraph ( )

Print Graph for user performance according to answering speed:

Print graphical characters for each result data in structure that holds record from result database updated by **NumberOfQuizTaken()** function according to its value.

Prompt for user input for exiting the function.

```
883 {   system("cls");
884 system("color bc");
885 system("title QuizBuddy-- Speed Graph");
886 int i,j;
887 int linesInGraph=15;
888 float factor= data.numberofQuizTaken/linesInGraph;
889 if ((float)factor<=1) factor=1;
890 if (linesInGraph>data.numberofQuizTaken) linesInGraph=data.numberofQuizTaken;
891 printf("\n\t\t\t   SPEED GRAPH\n\t\t\t_________________\n\n");
892 ///Print graphical characters for each result data in structure that holds record from result database
     updated by  NumberOfQuizTaken() function according to its value.
893 for(i=0; i<linesInGraph;i++)
894 {
895
896 int num=(int)(factor*i);
897 printf("%s%s",RsltFrmdtaStr[num].time, RsltFrmdtaStr[num].date);
898 {   for (j=0; j<(data.timeForeachQuestion-RsltFrmdtaStr[num].times/RsltFrmdtaStr[i].nOfQsns); j++)
899 if ((RsltFrmdtaStr[num].times/RsltFrmdtaStr[num].nOfQsns)<=data.timeForeachQuestion)
     printf("%c%c",0xDB,0xDB); printf("\n"); j++;
900 }
901 }
902 printf(" Date and Time\t  30        25        20        15        10        5        0\n");
903 printf(" The Lower Value indicate higher time spent on average.\n Higher the value, better is the
     result. \n");
904 printf("\n\t Press any Key to continue. . .");
905 ///Prompt for user input for exiting the function.
906 getch();
907 HomeScreen();
908 }
```

int QuizReview ( )

Let user review their answers after finishing quiz:

Prints the Questions, Options, Correct answers and your inputs for respective question during the last quiz.

Prompt the user character input for navigating through questions or exit.

Exit for respective user input.

```
{
    int i=0, nOfQsns= data.nOfQsns;  char get;
    while((get!='q')&&(get!='Q')&&(i>=0)&&(i<nOfQsns))
{
    ///Prints the Questions, Options, Correct answers and your inputs for respective question during the last quiz.
    system("cls");
    printf("\n\t\tQuiz Review\n\t\t__________\n");
printf("\n\t Question: %d of %d \t Percentage: %d\n", i+1,data.nOfQsns, data.lastQuizPerc);
    printf("\n\t %s\n", result[i].qsn);
        printf("\n\t Option 1: %s\n", result[i].ans1);
            printf("\n\t Option 2: %s\n", result[i].ans2);
                printf("\n\t Option 3: %s\n", result[i].ans3);
                    printf("\n\t Option 4: %s\n", result[i].ans4);
                        printf("\n\t  Correct Answer: %c\t\t Your Answer: %c\n", result[i].corans, result[i].userans);
                                if     (result[i].corans==result[i].userans)   {printf("\n\tYou answered correctly.\n\t\t__________\n"); system("@color A9");}
                                  else if (result[i].userans=='0') {printf("\n\tNo answer was chosen.\n\t\t__________\n"); system("@color F3");
                                   }
                                  else    {printf("\n\tYou answred wrong.\n\t\t__________\n");
                                  system("@color 0B");
                                   }
                                  printf("\n\tUse a and d to navigate and 'q' to quit review.");
                                  ///Prompt the user character input for navigating through questions or exit.
get=getch();
system("cls");
if ((get=='a')||(get=='A')) i--;
if ((get=='d')||(get=='D')) i++;
///Exit for respective user input.
if ((get=='q')||(get=='Q')) break;
if (i>nOfQsns-1) i--;
if (i<0) i++;
}
    printf("\n\t\tQuiz Review\n\t\t__________\n");
    printf("\n\t\tReview Completed. .  \n");
     return 0;
}
```

int QuizSelect (void)

This function allows user to select different quiz database. This function is followed by **download()** function that downloads the selected database from internet :

If file Quiz List not found, call to **firstrun()** .

Print Quiz List data from file.

Prompt for user input to select Quiz.

Set url and destination value as per user selection.

Call to **download()** .

```
{
    system("@cls");
    printf("\n Please Select your Quiz:\n\n");
       int end, loop;
    char str[600];
    FILE *database = fopen("QBcache/tslq.QBcache", "r");
    if (database == NULL)
    {       ///If file Quiz List not found, call to  firstrun() .
        printf("\n database null");
        firstrun();
        HomeScreen();
    }
    int line;
    ///Print Quiz List data from file.
    for (line=1;line<50;)
    {
    for(end = loop = 0;loop<line;++loop)
    {
            if(0==fgets(str, sizeof(str)+5, database)){
            end = 1;
            break;
        }
    }
    printf("%s", str);
    strcpy(str, "");

    if (end==1) break;
    line=2;
        }
    char usrQuizSelect[1];
    fflush(stdin);
    ///Prompt for user input to select Quiz.
    scanf("%s",&usrQuizSelect) ;
    line=(atoi(usrQuizSelect)+2)*2;
    rewind(database);  strcpy(str,"");
            for(end = loop = 0;loop<line;++loop){
        if(0==fgets(str, sizeof(str), database)){
            end = 1;
            break;
        }
    }
    ///Set url and destination value as per user selection.
        strcpy(url,""); strcpy(destination,""); strcpy(url,str);
    strcpy(destination,"QBcache/btd.QBcache" );

   fclose(database);

   ///Call to download() .
 if (download()==0)
{system("cls");
printf("\n\n Unsuccesful! Quiz Database was not downloaded. \n Try to update after internet
 connection.\n ");
sleep(5);
}
else HomeScreen();
            }
```

int QuizStart (int nOfQsns)

Function to Start Quiz :

Find no of question in database by counting number of lines; calling nOfQsnInDatabase() function.

Start Timer

Call the **datascanner()** with the random value from return of **RandomNumber()**, which lies in the range specified.

Print the Quiz Questions and answers.

Ask user inpur for correct answer.

Continuously Check for system time change comparing the take at beginning and at present.

If system time change detected, exit quiz showing fraud message.

Change the answer storing variable in result structure as per user answer.

Stop Timer

Check for correct answers and write update marks.

After finishing Asking Questions - Write Output to Data File

If file does not exist, create it

Display the result overview to user.

Calculate and Write percentage to structure variable data.

Write result to Data File.

Call the **QuizReview()** function to review the quiz after submission.

Prompt for user character input.

Goto **HomeScreen()** if user selects the respective option.

```
231 {
232     /// Find no of question in database by counting number of lines; calling nOfQsnInDatabase() function.
233 int nOfQsnInDatabase= findnOfQsnInDatabase("QBcache/btd.QBcache")-2;
234 ///Start Timer
235         clock_t qTimerStart = clock();
236 int i; int timeup;
237 for (i=0; i<(nOfQsns-1); i++)
238     {result[i].userans='0';
239     result[i].corans=0;
240     }
241    QsnNumbr=0; char get;
242 while((get!='q')&&(get!='Q'))
243 {
244     system("cls");
245     ///Call the datascanner() with the random value from return of RandomNumber(), which lies in the
     range specified.
246     RandArray[QsnNumbr]=(RandomNumber(nOfQsnInDatabase,QsnNumbr)+2);
247     if (result[QsnNumbr].corans==0) result[QsnNumbr].corans=datascanner(RandArray[QsnNumbr]);
248     ///Print the Quiz Questions and answers.
249     printf("\n\t\t    QuizBuddy -- Quiz Running \n");
250 printf("\n\t Question number: %d\t Number of Questions: %d\tTime: %.2fs\n\t\t__________\n",
     QsnNumbr+1, nOfQsns, (double)(clock() - qTimerStart) / CLOCKS_PER_SEC );
251 if (((double)(clock() - qTimerStart) / CLOCKS_PER_SEC )>= data.timeForeachQuestion*data.nOfQsns) {
     timeup=1; break;}
252     printf("\n\t %s\n", result[QsnNumbr].qsn);
253         printf("\n\t Option 1: %s\n", result[QsnNumbr].ans1);
254             printf("\n\t Option 2: %s\n", result[QsnNumbr].ans2);
255                 printf("\n\t Option 3: %s\n", result[QsnNumbr].ans3);
256                     printf("\n\t Option 4: %s\n", result[QsnNumbr].ans4);
257                         printf("\n\t Your Answer: %c\n",  result[QsnNumbr].userans);
258                               printf("\n\t\t__________\n\n\t PRESS 1,2,3,4 to CHANGE answer.\n\t PRESS
     a and d to NAVIGATE and PRESS O to SUBMIT answers.");
259 ///Ask user inpur for correct answer.
260 char get=getch();
261 ///Continuously Check for system time change comparing the take at beginning and at present.
262 if (((double)(clock() - qTimerStart) / CLOCKS_PER_SEC )<0)
263 {
264 system("cls");
265  printf("\n\tFraud Detected!\n\t_________________\n\n\tSystem Time Changed !!\n\tWe don't entertain
     that. :p \n\n\tBe fair next time %s .\n\tExiting QuizBuddy. . .. ", data.name);
266  sleep(3);
```

```
 ///If system time change detected, exit quiz showing fraud message.
exit(0);
  }
if ((get=='d')||(get=='D')) QsnNumbr++;
if ((get=='a')||(get=='A')) QsnNumbr--;
if ((get=='q')||(get=='Q')) break;
if ((get=='0') ||(get=='o')||(get=='O')) break;
///Change the answer storing variable in result structure as per user answer.
if (get=='1') {result[QsnNumbr].userans='1'; QsnNumbr++;}
if (get=='2') {result[QsnNumbr].userans='2'; QsnNumbr++;}
if (get=='3') {result[QsnNumbr].userans='3'; QsnNumbr++;}
if (get=='4') {result[QsnNumbr].userans='4'; QsnNumbr++;}
if (QsnNumbr>nOfQsns-1) QsnNumbr--;
if (QsnNumbr<0) QsnNumbr++ ;
system("cls");
}
    system("cls");
    printf("\n\t\tQuiz Completed\n\t\t__________\n");
    printf("\n\t\tSubmitting Answer. . \n");
sleep(2);

   /// Stop Timer
         clock_t qTimerEnd = clock();
         if (timeup==1) { system("cls"); printf("\n\t\tYou ran out of time. Be quick on next quiz.
 %s.\n", data.name);printf("\n\t\tGood Luck!!\n", data.name); sleep(4); qTimerEnd=
 qTimerStart+data.timeForeachQuestion*data.nOfQsns ; }

           for (i=0; i<(nOfQsns-1); i++)
           /// Check for correct answers and write update marks.
    {if (result[i].corans==result[i].userans) Marks++;
    }

    /// After finishing Asking Questions - Write Output to Data File
    FILE *ResultData;
  ResultData = fopen("QBcache/bdtsr.QBcache", "a");
  if(ResultData == NULL) { ResultData= fopen("QBcache/bdtsr.QBcache", "wb");  }  /// If file does not
 exist, create it
 ///Display the result overview to user.
 system("@cls");
 printf("\n\n \t RESULT\n\t_____\n\t_____\n\n");
printf("\tYou made %i out of %i questions correctly\n", Marks,nOfQsns) ;
printf("\tYou Took %.0f seconds to complete.\n", (double)(qTimerEnd - qTimerStart) / CLOCKS_PER_SEC);
printf("\tYour Accuracy Percentage: %d \n", (Marks*100/nOfQsns));
printf("\tYou answered one question in around %.0f second(s) on average. \n", (double)(qTimerEnd -
 qTimerStart) / (CLOCKS_PER_SEC*nOfQsns));
    if ((Marks*100/nOfQsns)>analysis.highestPrcntg) printf("\n\tHIGHEST RECORD!!\n");
        else if ((Marks*100/nOfQsns)>analysis.averageprcnt) printf("\n\tYou did better than your
 average records.\n");
        printf("\n\t_____\n\t");
        ///Calculate and Write percentage to structure variable data.
data.lastQuizPerc=(Marks*100/nOfQsns);
/// Write result to Data File.
fprintf(ResultData,"%d\t%d\t%f\t%s\t%s\n",nOfQsns ,Marks,((double)(qTimerEnd - qTimerStart) /
 CLOCKS_PER_SEC),__TIME__, __DATE__);
fclose(ResultData);
printf("\n\t%s\t%s\n",__DATE__,__TIME__);


      ///Call the QuizReview() function to review the quiz after submission.
printf("\n\n\n\t Press any key for QuizReview . . . \n\t Press q to skip  QuizReview.");
 ///Prompt for user character input.
 char tmp = getch();
if ((tmp=='q')||(tmp=='Q')) {HomeScreen(); }
      ///Goto HomeScreen() if user selects the respective option.
      //Call the QuizReview() function and then PrintProgressGraph();
      data.numberofQuizTaken=NumberOfQuizTaken();
      QuizReview();
      PrintProgressGraph();
  }
```

int RandomNumber (int max,
int i
)

Function to return a unique random number not present in array 'RandArray[100]' in a range value passed :

Loop until correct and unique value obtained.

Get a number time for process started and perform modulo operation by maximum value on the time.

Check if the value is in corect range and exclude false data.

Check if the value has previously been in RandArray[] , exclude data if exists.

If value is unique store the value in RandArray[] and return the value.

```
612 {
613     clock();
614 
615         int val=0;
616         int num;
617             ///Loop until correct and unique value obtained.
618         while (val!=5)
619         {
620             ///Get a number time for process started and perform modulo operation by maximum value on
    the time.
621                     num = clock()%max;
622             ///Check if the value is in corect range and exclude false data.
623             if  ((!(num>=minV&&num<=max))&(num==0)&(num==1)) {
624             val=4;
625                     }
626                     else val=5;
627             int j;
628             ///Check if the value has previously been in  RandArray[] , exclude data if exists.
629         for (j=0;j<i;j++)
630         {
631             if (num==RandArray[j]) {    val=4;  break;  }
632             else val=5;
633 
634             }
635 
636         }
637         ///If value is unique store the value in RandArray[] and return the value.
638             RandArray[i]= num;
639             return num;
640 
641     }
```

void settings (void)

This function takes the user to preferences section to change the program preferences :

If 1,2 or 3 Set no of question value acccording to option selected by user.

If 4, goto **firstrun()** .

If 5 goto **QuizSelect()** .

If 6 set url and destination to update program and call the **download()** .

If prompt for user confirmation and delete result data.

By default, return to HomeScreen for other key pressed.

```
{       //Get and check User Input for displayed preferences.
    system("@cls");
    system("@color fd");
    system("@title QuizBuddy -- Preferences ");
    printf("\n\t\tNumber of Questions Selected : %d\n\t\tAverage Time for each question: %d seconds.", data.nOfQsns, data.timeForeachQuestion);
    printf("\n\n\t\tPreferences\n\t\t_____________\n");
        printf("\n\t\tSelect Exam Type:\n\t\t1. 10 Questions\n\t\t2. 50 Questions\n\t\t3.100 Questions\n\n\t\t4. UpdateQuestionDatabase\n\t\t5. Quiz Selection\n\t\t6. Update QuizBuddy Program\n\t\t7. Reset Result Data");
    char settingType=getch();
    switch(settingType)
    ///If 1,2 or 3 Set no of question value acccording to option selected by user.
            {case '1':
    data.nOfQsns=10 ; break;
    case '2' :  data.nOfQsns=50; break;
    case '3' : data.nOfQsns=100; break;
    ///If 4, goto firstrun() .
    case '4' : firstrun(); break;
    ///If 5 goto QuizSelect() .
    case '5' : QuizSelect(); break;
        ///If 6 set url and destination to update program and call the download() .
    case '6' :  {
        strcpy(url,""); strcpy(destination,""); strcpy(url,"http://inter.net/somewhere/staticQuizList.xyz");
        strcpy(destination,"QuizBuddyUpdated.exe");
        if (download()==0)  HomeScreen();
                system("@ren QuizBuddy.exe QBold.exe");
                system("@ren QuizBuddyUpdated.exe QuizBuddy.exe");
                system("@move /y QBold.exe QBcache/");
                system("@del /Q /F QBold.exe");
                printf("\n\n\tQuizBuddy Updated\n\tPress any key . . .\n\t ");
                 getch();
                 printf("\n\n\tByeBye! This version of QuizBuddy will now exit.\n\tYou will meet the new and smarter QuizBuddy on next start.\n\t ");
                 sleep(5);
                 exit(0);
                  }
    case '7' :
        ///If  prompt for user confirmation and delete result data.
        {
    printf("\n\n\tYou are about to CLEAR all your result data.");
    printf("\n\n\tPress Y To CONFIRM or any key to cancel.\n");
    char tmp=getch();
    if ((tmp=='y')||(tmp=='Y'))
    {
remove("QBcache/bdtsr.QBcache");
printf("\n\n\tResult Cleared!!\n");
data.lastQuizPerc=0;
   sleep(2);
        main();
    }


}
///By default, return to HomeScreen for other key pressed.
    default : system("@cls");   HomeScreen();
}
}
```

char UserRecord (void)

This function edits personal information record of user:

Ask the user informations.

Store it in data structure which is written to record file.

Display the user information for verification.

Return to main menu

```
690 {
691         system("@title QuizBuddy -- Profile Edit ");
692         ///Ask the user informations.
693          printf("\n\tPlease edit your profile:\n\t\t");
694     printf("\n\tInput your full name:\n\t\t");
695      fflush(stdin);
696      ///Store it in data structure which is written to record file.
697      gets(data.name);
698       printf("\n\tInput your email address:\n\t\t");
699      gets(data.address);
700      printf("\n\tInput associated institution name:\n\t\t");
701      gets(data.collegename);
702      printf("\n\tInput the name of exam you are preparing for:\n\t\t");
703      gets(data.preparingfor);
704      printf("\n\tInput contact number:\n\t");
705      gets(data.mobilenum);
706      printf("\n\tInput your age in number:\n\t\t");
707      scanf("%d", &data.age);
708      fflush(stdin);
709     system("cls");
710     ///Display the user information for verification.
711      printf("\n\tPROFILE REVIEW\n\t________________\n");
712      printf("\n\t Name: %s ",data.name  );
713     printf("\n\t Age: %ld ",data.age  );
714     printf("\n\t Email : %s ",data.address  );
715     printf("\n\t Institution name: %s ",data.collegename  );
716     printf("\n\t Exam preparing for: %s ",data.preparingfor  );
717      printf("\n\t Contact Number: %s ",data.mobilenum  );
718      FILE *userrec = fopen("QBcache/tdrsu.QBcache", "wb");
719        fwrite(&data, sizeof(data),1,userrec);
720     fclose(userrec);
721      printf("\n\n\t\tYour Profile has been updated.\n\t\tYou can always edit your profile..", data.name);
722      printf("\n\tReturning Back to Main Menu. . " );
723     ///Return to main menu
724      sleep(5);
725 }
```

void viewprofile (void)

This funcction displays personal information record of user:

Display the user information for verification.

Prompt user character input for result view or profile edit or clearing data.

Perform the task as per user input : result display through **printQuizData()** or profile edit through **UserRecord()**.

Return to **HomeScreen()**.

```
{system("@title QuizBuddy -- Profile review ");
system("cls");
  ///Display the user information for verification.
      printf("\n\tPROFILE REVIEW\n\t_________________\n");
      printf("\n\n\t Name: %s ",data.name  );
     printf("\n\t Age: %ld ",data.age  );
     printf("\n\t Email : %s ",data.address  );
     printf("\n\t Institution name: %s ",data.collegename  );
     printf("\n\t Exam preparing for: %s ",data.preparingfor  );
      printf("\n\t Contact Number: %s \n\n\t_________________\n",data.mobilenum  );

      printf("\n\n\tPress r key from keyboard to view your QUIZ RESULTS and GRAPHS");
      printf("\n\n\tPress e key from keyboard to EDIT your PROFILE");
      printf("\n\n\tPress l key from keyboard to LOG OUT");
      char tmp= getch();
            ///Prompt user character input for result view or profile edit or clearing data.
            system("cls");
          ///Perform the task as per user input : result display through printQuizData() or profile edit through UserRecord().
     if ((tmp==('e'))||(tmp==('E'))) {UserRecord();}
     if ((tmp==('r'))||(tmp==('R'))) {printQuizData();}
     if ((tmp=='l')||(tmp=='L'))       {
    printf("\n\n\tYou are about to log out ");
    printf("\n\n\t and CLEAR all your personal data including results.");
    printf("\n\n\tPress Y To CONFIRM or any key to cancel.\n");
    char tmp=getch();
    if ((tmp=='y')||(tmp=='Y'))
    {
remove("QBcache/tdrsu.QBcache");
remove("QBcache/bdtsr.QBcache");
   sleep(2);
    main();
    }

     }
      system("cls");
      ///Return to HomeScreen().
     HomeScreen();
}
```

Variable Documentation

struct UserResultAnalysis analysis

Structure variable for **UserResultAnalysis**. This structure holds user performance informations.

struct UserRecSt data

Structure variable for **UserRecSt**. This structure is saved to record file and data is retrieved on next program start.

char destination[MAX_LINE]

Download destination variable used by **download()** function.

int Marks =0

Reset marks variable to zero.

int QsnNumbr =0

Reset Question number variable to zero:

int RandArray[100]

Integer array to store 100 unique random numbers in a range generated by RandomNumber function.

struct resultData result[100]

Structure variable for **resultData** to hold values of question and answers along with correct answer while using **QuizStart()**;.

struct ResultFromdataStr RsltFrmdtaStr[99999]

Structure variable for **ResultFromdataStr** to hold the values of data retrieved from result database.

char str[512]

String variable to store temporary data during file reading .

char url[MAX_LINE]

Download Url variable used by **download()** function.