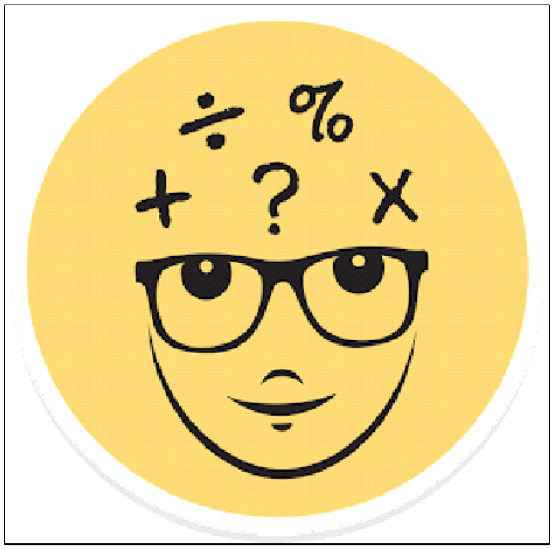


Mini Project Documentation

MATH QUIZ GAME

Institute of Applied Medicines and

Research

GHAZIABAD

CERTIFICATE

This report on "**MATH QUIZ GAME**" is a bonafide record of

the Minor-project work submitted By

AYUSHI SINGHAL (ROLL NO. 1709661766)

HARDIK KANSAL (ROLL NO.1709661778)

MANISH SINGH MEHRA (ROLL NO. 1709661796)

MOH.SHAHRUKH KHAN (ROLL NO.1709661798)

In their fifth semester of BCA

(Bachelor of computer application)

During the academic year 2017-2020

Guide
Observer

HEAD OF DEPARTMENT

Candidate's Declaration

We hereby declare that the work presented in this project titled "MATH QUIZ GAME" submitted towards completion of minor-project in

fifth semester of BCA at the **Institute of Applied Medicines and Research , GZB** is an authentic

record of our original work pursed under the guidance of **Prof.TRIBHUWAN TYAGI SIR** ,IAMR GZB.

We have not submitted the matter embodied in this project for the award of any degree.

AYUSHI SINGHAL

HARDIK KANSAL

MANISH MEHRA

MOH.SHAHRUKH

Place: Ghaziabad

Date:

ACKNOWLEDGMENT

First and foremost, we would like to express our sincere gratitude to our minor project guides, Prof.Tribhuwan tyagi . we were privileged to experience a substained enthusiastic and involved interest from their side. This fueled our enthusiasm even further and encouraged us to boldly step into what was a totally dark and unexplored expanse before us.

ABSTRACT

INTRODUCTION

A quiz is a form of game or mind sport, in which the players (as individuals or in teams) attempt to answers questions correctly. it is a game to test the knowledge about a certain subject. In some countries, a quiz is also a brief assessment used in education and similar fields to measure growth in knowledge, abilities, and/or skills.

In an educational context, a quiz is usually a form of a student assessment, but often has fewer questions of less difficulty and requires less time for completion than a test.

Math quiz helps us to increase our knowledge.

Online math quizzes will take 5 minutes of your time to complete a set of questions on math test quiz which will help you to know how much you know about math quizzes and how much time you need to complete a set of math questions. If you want you can also tale printable math quizzes from your home printers and share the questions with your friends.

MOTIVATION

Math Games + Fun = Learning

Math games make learning Math fun. when we're having fun, we're more open to learning. when we're having fun, we wants to keep doing whatever we're doing.

So if you're a parent, teacher, or student and want to make math more user-friendly, come and explore our site where you'll find lots of fun math activities and other cool math stuff!

PROBLEM STATEMENT

You sit at your desk, ready to put a math quiz, test or activity together. The questions flow onto the document until you

hit a section for word problem.

A jolt of creativity would help. But it doesn't come.

This resource is your jolt of creativity. it provides examples and templates of math problems for 1st to 8th grade classes.

Relevance :

Quiz Game is an application that has general questions related to simple calculations and logical questions. It has multiple choice questions with time limit and it also calculate scores of each correct answer. It is good for students of every age group it helps in increasing about basic calculations, mathematical formulas, logical questions etc. Don't need register simply give any user name and password it will saved automatically and you can login again with same user name and password don't have to worry about the past score. The application helps the user to increase

his/her knowledge. Since smartphone mobiles are being widely used by general popoulation and students, the quiz contest application can provide on the student's mobile.

Problem Definition:

Quiz Game is a application developed to conduct an quiz based on time constraints. Quiz Game system is accessed by entering the user name and password which is added to the database. Before start of the quiz, the rules and regulations are displayed that includes description of the time limit, number of questions to be answered and scoring methods. Quiz is started by displaying one question with four options each based on simple calculations and logical questions. If the answer is correct, score is incremented by four and no negative marks for wrong answers. And after completing all the questions application will finally direct you to the score page. Final score will be displayed and updated in the database

username.

Objective:

The main objective of "QUIZ GAME" is to facilitate a user friendly environment for all users and reduces the manual effort. In past days quiz is conducted manually but in future resolution of the technology we are able to generate the score and pose the queries automatically. The functional requirements include to create users that are going to participate in the Quiz, automatic score and report generation and administrative tasks like add, delete, update for admin privilege users. In this application, all the permissions lies with administrator i.e., specifying the details of the quiz with checking result will show to interviewer or not, addition of question and answers, marks for each questions, Set timer for each quiz and generate report with score for each quiz.

BASIC CONCEPTS & TOOLS :

INTRODUCTION

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. It was created by Guido van Rossum during 1985- 1990. Like Perl, Python source code is also available under the GNU General Public License (GPL). This tutorial gives enough understanding on Python programming language.

Why to Learn Python?

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

Python is a MUST for students and working professionals to become a great Software Engineer specially when they are working in Web Development Domain. I will list down some of the key advantages of learning Python:

- **Python is Interpreted** − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.

- **Python is Interactive** − You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

- **Python is Object-Oriented** − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.

- **Python is a Beginner's Language** – Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

Characteristics of Python

Following are important characteristics of Python Programming –

- It supports functional and structured programming methods as well as OOP.

- It can be used as a scripting language or can be compiled to byte-code for building large applications.

- It provides very high-level dynamic data types and supports dynamic type checking.

- It supports automatic garbage collection.

- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

Hello World using Python.

Just to give you a little excitement about Python, I'm going to give you a small conventional Python Hello World program.

```
print ("Hello, Python!")
```

Applications of Python

As mentioned before, Python is one of the most widely used language over the web. I'm going to list few of them here:

- **Easy-to-learn** – Python has few keywords, simple structure, and a clearly defined

syntax. This allows the student to pick up the language quickly.

- **Easy-to-read** − Python code is more clearly defined and visible to the eyes.

- **Easy-to-maintain** − Python's source code is fairly easy-to-maintain.

- **A broad standard library** − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.

- **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.

- **Portable** − Python can run on a wide variety of hardware platforms and has the same

interface on all platforms.

- **Extendable** – You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.

- **Databases** – Python provides interfaces to all major commercial databases.

- **GUI Programming** – Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.

- **Scalable** – Python provides a better structure and support for large programs than shell scripting.

Prerequisites

You should have a basic understanding of Computer Programming terminologies. A basic understanding of any of the programming languages is a plus.

History of Python

- Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

- Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.

- Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

- Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

Python - Modules

A module allows you to logically organize your Python code. Grouping related code into a module makes the code easier to understand and use. A module is a Python object with arbitrarily named attributes that you can bind and reference.

Simply, a module is a file consisting of Python code. A module can define functions, classes and variables. A module can also include runnable code.

Example

The Python code for a module named aname normally resides in a file named aname.py. Here's an example of a simple module, support.py

```
def print_func( par ):
    print ("Hello : ", par)
    return
```

The import Statement

You can use any Python source file as a

module by executing an import statement in some other Python source file. The import has the following **syntax –**

import module1[, module2[,... moduleN]

When the interpreter encounters an import statement, it imports the module if the module is present in the search path. A search path is a list of directories that the interpreter searches before importing a module. For example, to import the module support.py, you need to put the following command at the top of the script .

#!/usr/bin/python

 # Import module support

import support

 # Now you can call defined function that module as follows

support.print_func("Zara")

When the above code is executed, it produces the following result –

Hello : Zara

A module is loaded only once, regardless of the number of times it is imported. This prevents the module execution from happening over and over again if multiple imports occur.

The from...import Statement

Python's from statement lets you import specific attributes from a module into the

current namespace. The from...import has the following syntax −

from modname import name1[, name2[, ... nameN]]

For example, to import the function fibonacci from the module fib, use the following statement −

from fib import fibonacci

This statement does not import the entire module fib into the current namespace; it just introduces the item fibonacci from the module fib into the global symbol table of the importing module.

The from...import * Statement

It is also possible to import all names from

a module into the current namespace by using the following import statement −

from modname import *

This provides an easy way to import all the items from a module into the current namespace; however, this statement should be used sparingly.

Locating Modules

When you import a module, the Python interpreter searches for the module in the following sequences −

The current directory.

If the module isn't found, Python then

searches each directory in the shell variable PYTHONPATH.

If all else fails, Python checks the default path. On UNIX, this default path is normally /usr/local/lib/python/.

The module search path is stored in the system module sys as the sys.path variable. The sys.path variable contains the current directory, PYTHONPATH, and the installation-dependent default.

The PYTHONPATH Variable

The PYTHONPATH is an environment variable, consisting of a list of directories. The syntax of PYTHONPATH is the same as that of the shell variable PATH.

Here is a typical PYTHONPATH from a Windows system −

```
set PYTHONPATH = c:\python20\lib;
```

And here is a typical PYTHONPATH from a UNIX system −

```
set PYTHONPATH = /usr/local/lib/python
```

Namespaces and Scoping

Variables are names (identifiers) that map to objects. A namespace is a dictionary of variable names (keys) and their corresponding objects (values).

A Python statement can access variables in a local namespace and in the global namespace. If a local and a global variable have the same name, the local variable

shadows the global variable.

Each function has its own local namespace. Class methods follow the same scoping rule as ordinary functions.

Python makes educated guesses on whether variables are local or global. It assumes that any variable assigned a value in a function is local.

Therefore, in order to assign a value to a global variable within a function, you must first use the global statement.

The statement global VarName tells Python that VarName is a global variable. Python stops searching the local namespace for the variable.

For example, we define a variable Money in the global namespace. Within the function Money, we assign Money a value, therefore Python assumes Money as a local variable. However, we accessed the value of the local variable Money before setting it, so an UnboundLocalError is the result. Uncommenting the global statement fixes the problem.

```
#!/usr/bin/python

Money = 2000
def AddMoney():
    # Uncomment the following line to fix the code:
    # global Money
```

```
    Money = Money + 1

print Money
AddMoney()
print Money
```

Python - GUI Programming (Tkinter)

Python provides various options for developing graphical user interfaces (GUIs). Most important are listed below.

- **Tkinter** – Tkinter is the Python interface to the Tk GUI toolkit shipped with Python. We would look this option in this chapter.

- **wxPython** – This is an open-source Python interface for wxWindows http://wxpython.org.

- **JPython** – JPython is a Python port for Java which gives Python scripts seamless access to Java class libraries on the local machine http://www.jython.org.

There are many other interfaces available, which you can find them on the net.

Tkinter Programming

Tkinter is the standard GUI library for

Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.

Creating a GUI application using Tkinter is an easy task. All you need to do is perform the following steps –

Import the Tkinter module.

Create the GUI application main window.

Add one or more of the above-mentioned widgets to the GUI application.

Enter the main event loop to take action

against each event triggered by the user.

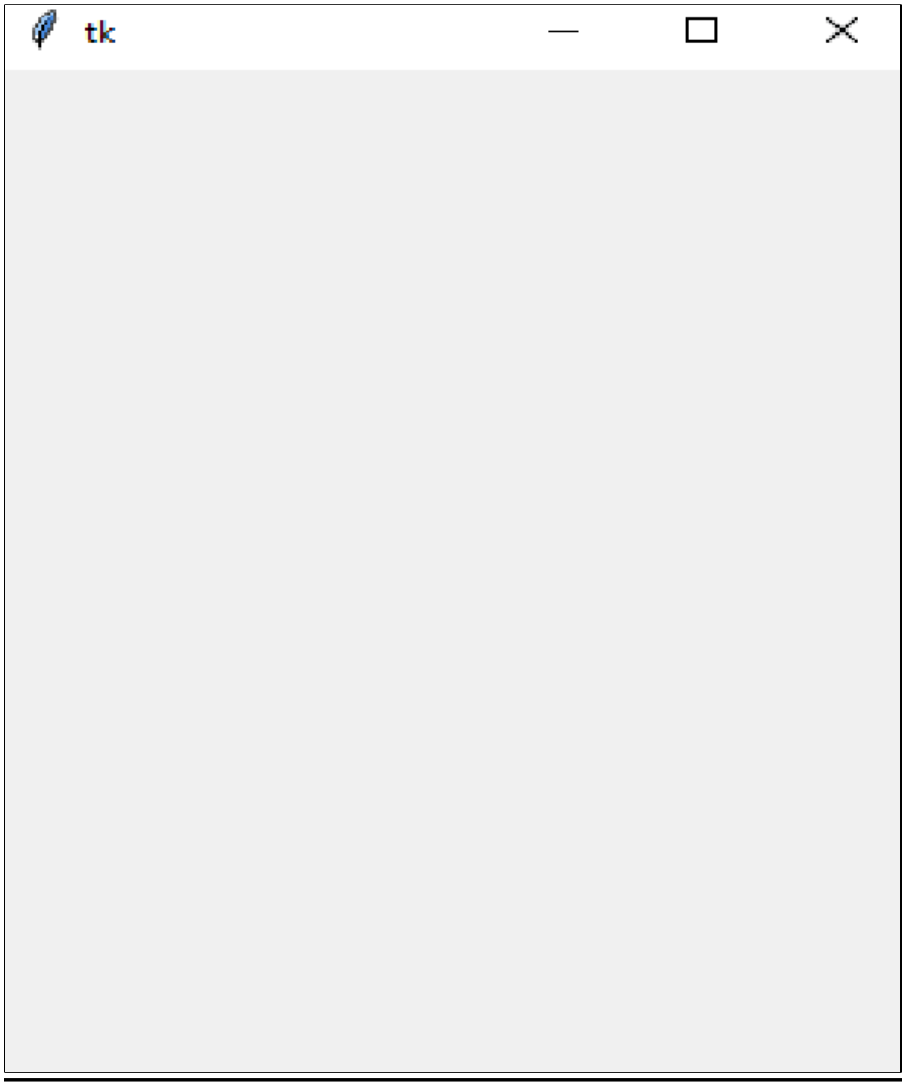
Example

```
#!/usr/bin/python

import Tkinter
top = Tkinter.Tk()
# Code to add widgets will go here...
top.mainloop()
```

This would create a following window –

TK Window



Tkinter Widgets

Tkinter provides various controls, such as

buttons, labels and text boxes used in a GUI application. These controls are commonly called widgets.

There are currently 15 types of widgets in Tkinter. We present these widgets as well as a brief description in the following table –

1 Button

The Button widget is used to display buttons in your application.

2 Canvas

The Canvas widget is used to draw shapes, such as lines, ovals, polygons and rectangles, in your application.

3 Checkbutton

The Checkbutton widget is used to display a number of options as checkboxes. The user can select multiple options at a time.

4 Entry

The Entry widget is used to display a single-line text field for accepting values from a user.

5 Frame

The Frame widget is used as a container widget to organize other widgets.

6 Label

The Label widget is used to provide a

single-line caption for other widgets. It can also contain images.

7 Listbox

The Listbox widget is used to provide a list of options to a user.

8 Menubutton

The Menubutton widget is used to display menus in your application.

9 Menu

The Menu widget is used to provide various commands to a user. These commands are contained inside Menubutton.

10 Message

The Message widget is used to display multiline text fields for accepting values from a user.

11 Radiobutton

The Radiobutton widget is used to display a number of options as radio buttons. The user can select only one option at a time.

12 Scale

The Scale widget is used to provide a slider widget.

13 Scrollbar

The Scrollbar widget is used to add

scrolling capability to various widgets, such as list boxes.

14 Text

The Text widget is used to display text in multiple lines.

15 Toplevel

The Toplevel widget is used to provide a separate window container.

16 Spinbox

The Spinbox widget is a variant of the standard Tkinter Entry widget, which can be used to select from a fixed number of values.

17 PanedWindow

A PanedWindow is a container widget that may contain any number of panes, arranged horizontally or vertically.

18 LabelFrame

A labelframe is a simple container widget. Its primary purpose is to act as a spacer or container for complex window layouts.

19 tkMessageBox

This module is used to display message boxes in your applications.

Let us study these widgets in detail –

Standard attributes

Let us take a look at how some of their common attributes.such as sizes, colors and fonts are specified.

- **Dimensions**
- **Colors**
- **Fonts**
- **Anchors**
- **Relief styles**
- **Bitmaps**
- **Cursors**

Let us study them briefly –

Geometry Management

All Tkinter widgets have access to specific geometry management methods, which have the purpose of organizing widgets throughout the parent widget area. Tkinter exposes the following geometry manager classes: pack, grid, and place.

- **The pack() Method –** This geometry manager organizes widgets in blocks before placing them in the parent widget.

- **The grid() Method –** This geometry manager organizes widgets in a table-like structure in the parent widget.

- **The place() Method –** This geometry manager organizes widgets by placing them in a specific position in the parent widget.

Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is the most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter is the fastest and easiest way to create the GUI applications. Creating a GUI using tkinter is an easy task.

To create a tkinter app:

1. Importing the module — tkinter
2. Create the main window (container)
3. Add any number of widgets to the

main window

4. Apply the event Trigger on the widgets.

Importing tkinter is same as importing any other module in the Python code. Note that the name of the module in Python 2.x is Tkinter and in Python 3.x it is tkinter.

import tkinter

There are two main methods used which the user needs to remember while creating the Python application with GUI.

1. Tk(screenName=None, baseName=None, className=Tk, useTk=1): To create a main window, tkinter offers a method Tk(screenName=None, baseName=None, className=Tk,

useTk=1). To change the name of the window, you can change the className to the desired one. The basic code used to create the main window of the application is:

m=tkinter.Tk() where m is the name of the main window object

2. mainloop(): There is a method known by the name mainloop() is used when your application is ready to run. mainloop() is an infinite loop used to run the application, wait for an event to occur and process the event as long as the window is not closed.

m.mainloop()

filter_none

brightness_4

import tkinter

m = tkinter.Tk()

widgets are added here

m.mainloop()

tkinter also offers access to the geometric configuration of the widgets which can organize the widgets in the parent windows. There are mainly three geometry manager classes class.

pack() method: It organizes the widgets in blocks before placing in the parent widget.

grid() method: It organizes the widgets in grid (table-like structure) before placing in the parent widget.

place() method: It organizes the widgets by placing them on specific positions directed by the programmer.

There are a number of widgets which you can put in your tkinter application. Some of the major widgets are explained below:

Button: To add a button in your application, this widget is used.

The general syntax is:

w=Button(master, option=value)

master is the parameter used to represent the parent window.

There are number of options which are used to change the format of the Buttons. Number of options can be passed as parameters separated by commas. Some of them are listed below.

- activebackground: to set the background color when button is under the cursor.

- activeforeground: to set the foreground color when button is under the cursor.
- bg: to set he normal background color.
- command: to call a function.
- font: to set the font on the button label.
- image: to set the image on the button.
- width: to set the width of the button.
- height: to set the height of the button.

```
import tkinter as tk

r = tk.Tk()

r.title('Counting Seconds')

button = tk.Button(r, text='Stop', width=25, command=r.destroy)

button.pack()

r.mainloop()
```

Canvas: It is used to draw pictures and other complex layout like graphics, text and widgets.

The general syntax is:

w = Canvas(master, option=value)

master is the parameter used to represent the parent window.

There are number of options which are used to change the format of the widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

- bd: to set the border width in pixels.
- bg: to set the normal background color.
- cursor: to set the cursor used in the canvas.
- highlightcolor: to set the color shown in

the focus highlight.

- width: to set the width of the widget.
- height: to set the height of the widget.

```
from tkinter import *
master = Tk()
w = Canvas(master, width=40, height=60)
w.pack()
canvas_height=20
canvas_width=200
y = int(canvas_height / 2)
w.create_line(0, y, canvas_width, y )
mainloop()
```

CheckButton: To select any number of options by displaying a number of options to a user as toggle buttons. The general syntax is:

w = CheckButton(master, option=value)

There are number of options which are used to change the format of this widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

- Title: To set the title of the widget.
- activebackground: to set the background color when widget is under the cursor.
- activeforeground: to set the foreground color when widget is under the cursor.
- bg: to set he normal backgrouSteganography
- Break

Secret Code:

Attach a File:nd color.

- command: to call a function.
- font: to set the font on the button label.
- image: to set the image on the widget.
- filter_none
- edit
- play_arrow

```
from tkinter import *
master = Tk()
var1 = IntVar()
Checkbutton(master, text='male', variable=var1).grid(row=0, sticky=W)
var2 = IntVar()
Checkbutton(master, text='female',
```

variable=var2).grid(row=1, sticky=W)

mainloop()

Entry: It is used to input the single line text entry from the user.. For multi-line text input, Text widget is used.

The general syntax is:

w=Entry(master, option=value)

master is the parameter used to represent the parent window.

There are number of options which are used to change the format of the widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

- bd: to set the border width in pixels.

- bg: to set the normal background color.
- cursor: to set the cursor used.
- command: to call a function.
- highlightcolor: to set the color shown in the focus highlight.
- width: to set the width of the button.
- height: to set the height of the button.
- filter_none
- edit
- play_arrow

```
from tkinter import *
master = Tk()
Label(master, text='First Name').grid(row=0)
Label(master, text='Last Name').grid(row=1)
e1 = Entry(master)
```

```
e2 = Entry(master)
e1.grid(row=0, column=1)
e2.grid(row=1, column=1)
mainloop()
```

Frame: It acts as a container to hold the widgets. It is used for grouping and organizing the widgets. The general syntax is:

```
w = Frame(master, option=value)
```

master is the parameter used to represent the parent window.

There are number of options which are used to change the format of the widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

- highlightcolor: To set the color of the focus highlight when widget has to be focused.
- bd: to set the border width in pixels.
- bg: to set the normal background color.
- cursor: to set the cursor used.
- width: to set the width of the widget.
- height: to set the height of the widget.
- filter_none
- edit
- play_arrow
- brightness_4

```
from tkinter import *
root = Tk()
frame = Frame(root)
frame.pack()
```

```
bottomframe = Frame(root)

bottomframe.pack( side = BOTTOM )

redbutton = Button(frame, text = 'Red', fg ='red')

redbutton.pack( side = LEFT)

greenbutton = Button(frame, text = 'Brown', fg='brown')

greenbutton.pack( side = LEFT )

bluebutton = Button(frame, text ='Blue', fg ='blue')

bluebutton.pack( side = LEFT )

blackbutton = Button(bottomframe, text ='Black', fg ='black')

blackbutton.pack( side = BOTTOM)

root.mainloop()
```

Label: It refers to the display box where you

can put any text or image which can be updated any time as per the code.

The general syntax is:

w=Label(master, option=value)

master is the parameter used to represent the parent window.

There are number of options which are used to change the format of the widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

- bg: to set he normal background color.
- bg to set he normal background color.
- command: to call a function.
- font: to set the font on the button label.
- image: to set the image on the button.

- width: to set the width of the button.
- height to set the height of the button.
- filter_none
- edit
- play_arrow
- brightness_4

```
from tkinter import *
root = Tk()
w = Label(root, text='GeeksForGeeks.org!')
w.pack()
root.mainloop()
```

Listbox: It offers a list to the user from which the user can accept any number of options.

The general syntax is:

w = Listbox(master, option=value)

master is the parameter used to represent the parent window.

There are number of options which are used to change the format of the widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

- highlightcolor: To set the color of the focus highlight when widget has to be focused.
- bg: to set he normal background color.
- bd: to set the border width in pixels.
- font: to set the font on the button label.
- image: to set the image on the widget.
- width: to set the width of the widget.
- height: to set the height of the widget.

- filter_none
- edit
- play_arrow
- brightness_4

```
from tkinter import *
top = Tk()
Lb = Listbox(top)
Lb.insert(1, 'Python')
Lb.insert(2, 'Java')
Lb.insert(3, 'C++')
Lb.insert(4, 'Any other')
Lb.pack()
top.mainloop()
```

MenuButton: It is a part of top-down menu which stays on the window all the time.

Every menubutton has its own functionality. The general syntax is:

w = MenuButton(master, option=value)

master is the parameter used to represent the parent window.

There are number of options which are used to change the format of the widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

- activebackground: To set the background when mouse is over the widget.
- activeforeground: To set the foreground when mouse is over the widget.
- bg: to set he normal background color.
- bd: to set the size of border around the indicator.

- cursor: To appear the cursor when the mouse over the menubutton.
- image: to set the image on the widget.
- width: to set the width of the widget.
- height: to set the height of the widget.
- highlightcolor: To set the color of the focus highlight when widget has to be focused.
- filter_none
- edit
- play_arrow
- brightness_4

```
from tkinter import *

top = Tk()

mb = Menubutton ( top, text = &quot;GfG&quot;)

mb.grid()
```

```
mb.menu    =    Menu ( mb, tearoff = 0 )

mb[&quot;menu&quot;]    =    mb.menu

cVar    = IntVar()

aVar = IntVar()

mb.menu.add_checkbutton ( label ='Contact', variable = cVar )

mb.menu.add_checkbutton ( label = 'About', variable = aVar )

mb.pack()

top.mainloop()
```

Menu: It is used to create all kinds of menus used by the application.

The general syntax is:

w = Menu(master, option=value)

master is the parameter used to represent the parent window.

There are number of options which are used to change the format of this widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

- title: To set the title of the widget.
- activebackground: to set the background color when widget is under the cursor.
- activeforeground: to set the foreground color when widget is under the cursor.
- bg: to set he normal background color.
- command: to call a function.
- font: to set the font on the button label.
- image: to set the image on the widget.
- filter_none
- edit

- play_arrow
- brightness_4
- from tkinter import *
- root = Tk()
- menu = Menu(root)
- root.config(menu=menu)
- filemenu = Menu(menu)
- menu.add_cascade(label='File', menu=filemenu)
- filemenu.add_command(label='New')
- filemenu.add_command(label='Open...')
- filemenu.add_separator()
- filemenu.add_command(label='Exit', command=root.quit)
- helpmenu = Menu(menu)
- menu.add_cascade(label='Help',

menu=helpmenu)

- helpmenu.add_command(label='About')
- mainloop()

Message: It refers to the multi-line and non-editable text. It works same as that of Label.

The general syntax is:

w = Message(master, option=value)

master is the parameter used to represent the parent window.

There are number of options which are used to change the format of the widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

- bd: to set the border around the

indicator.

- bg: to set he normal background color.
- font: to set the font on the button label.
- image: to set the image on the widget.
- width: to set the width of the widget.
- height: to set the height of the widget.
- filter_none
- edit
- play_arrow
- brightness_4

```
from tkinter import *
main = Tk()
ourMessage ='This is our Message'
messageVar = Message(main, text = ourMessage)
messageVar.config(bg='lightgreen')
```

messageVar.pack( )

main.mainloop( )

RadioButton: It is used to offer multi-choice option to the user. It offers several options to the user and the user has to choose one option.

The general syntax is:

w = RadioButton(master, option=value)

There are number of options which are used to change the format of this widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

- activebackground: to set the background color when widget is under the cursor.

- activeforeground: to set the foreground color when widget is under the cursor.
- bg: to set he normal background color.
- command: to call a function.
- font: to set the font on the button label.
- image: to set the image on the widget.
- width: to set the width of the label in characters.
- height: to set the height of the label in characters.
- filter_none
- edit
- play_arrow
- brightness_4

```
from tkinter import *
root = Tk()
```

```
v = IntVar()

Radiobutton(root, text='GfG', variable=v,
value=1).pack(anchor=W)

Radiobutton(root, text='MIT', variable=v,
value=2).pack(anchor=W)

mainloop()
```

Scale: It is used to provide a graphical slider that allows to select any value from that scale. The general syntax is:

```
w = Scale(master, option=value)
```

master is the parameter used to represent the parent window.

There are number of options which are used to change the format of the widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

- cursor: To change the cursor pattern when the mouse is over the widget.
- activebackground: To set the background of the widget when mouse is over the widget.
- bg: to set he normal background color.
- orient: Set it to HORIZONTAL or VERTICAL according to the requirement.
- from_: To set the value of one end of the scale range.
- to: To set the value of the other end of the scale range.
- image: to set the image on the widget.
- width: to set the width of the widget.
- filter_none
- edit

- play_arrow
- brightness_4

```
from tkinter import *
master = Tk()
w = Scale(master, from_=0, to=42)
w.pack()
w = Scale(master, from_=0, to=200, orient=HORIZONTAL)
w.pack()
mainloop()
```

Scrollbar: It refers to the slide controller which will be used to implement listed widgets.

The general syntax is:

w = Scrollbar(master, option=value)

master is the parameter used to represent

the parent window.

There are number of options which are used to change the format of the widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

- width: to set the width of the widget.
- activebackground: To set the background when mouse is over the widget.
- bg: to set he normal background color.
- bd: to set the size of border around the indicator.
- cursor: To appear the cursor when the mouse over the menubutton.
- filter_none
- edit

- play_arrow
- brightness_4

```
from tkinter import *
root = Tk()
scrollbar = Scrollbar(root)
scrollbar.pack( side = RIGHT, fill = Y )
mylist = Listbox(root, yscrollcommand = scrollbar.set )
for line in range(100):
    mylist.insert(END, 'This is line number' + str(line))
mylist.pack( side = LEFT, fill = BOTH )
scrollbar.config( command = mylist.yview )
mainloop()
```

Text: To edit a multi-line text and format the way it has to be displayed.

The general syntax is:

w =Text(master, option=value)

There are number of options which are used to change the format of the text. Number of options can be passed as parameters separated by commas. Some of them are listed below.

- highlightcolor: To set the color of the focus highlight when widget has to be focused.
- insertbackground: To set the background of the widget.
- bg: to set he normal background color.
- font: to set the font on the button label.
- image: to set the image on the widget.
- width: to set the width of the widget.
- height: to set the height of the widget.

- filter_none
- edit
- play_arrow
- brightness_4

```
from tkinter import *

root = Tk()

T = Text(root, height=2, width=30)

T.pack()

T.insert(END, 'GeeksforGeeks\nBEST WEBSITE\n')

mainloop()
```

TopLevel: This widget is directly controlled by the window manager. It dont need any parent window to work on.The general syntax is:

```
w = TopLevel(master, option=value)
```

There are number of options which are used to change the format of the widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

- bg: to set he normal background color.
- bd: to set the size of border around the indicator.
- cursor: To appear the cursor when the mouse over the menubutton.
- width: to set the width of the widget.
- height: to set the height of the widget.
- filter_none
- edit
- play_arrow
- brightness_4

```
from tkinter import *
root = Tk()
root.title('GfG')
top = Toplevel()
top.title('Python')
top.mainloop()
```

SpinBox: It is an entry of Entry widget. Here, value can be input by selecting a fixed value of numbers.The general syntax is:

```
w = SpinBox(master, option=value)
```

There are number of options which are used to change the format of the widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

- bg: to set he normal background color.

- bd: to set the size of border around the indicator.
- cursor: To appear the cursor when the mouse over the menubutton.
- command: To call a function.
- width: to set the width of the widget.
- activebackground: To set the background when mouse is over the widget.
- disabledbackground: To disable the background when mouse is over the widget.
- from_: To set the value of one end of the range.
- to: To set the value of the other end of the range.
- filter_none
- edit

- play_arrow
- brightness_4

```
from tkinter import *
master = Tk()
w = Spinbox(master, from_ = 0, to = 10)
w.pack()
mainloop()
```

PannedWindow: It is a container widget which is used to handle number of panes arranged in it. The general syntax is:

```
w = PannedWindow(master, option=value)
```

master is the parameter used to represent the parent window.

There are number of options which are used to change the format of the widget. Number of options can be passed as parameters separated by commas. Some of them are

listed below.

- bg: to set he normal background color.
- bd: to set the size of border around the indicator.
- cursor: To appear the cursor when the mouse over the menubutton.
- width: to set the width of the widget.
- height: to set the height of the widget.
- filter_none
- edit
- play_arrow
- brightness_4

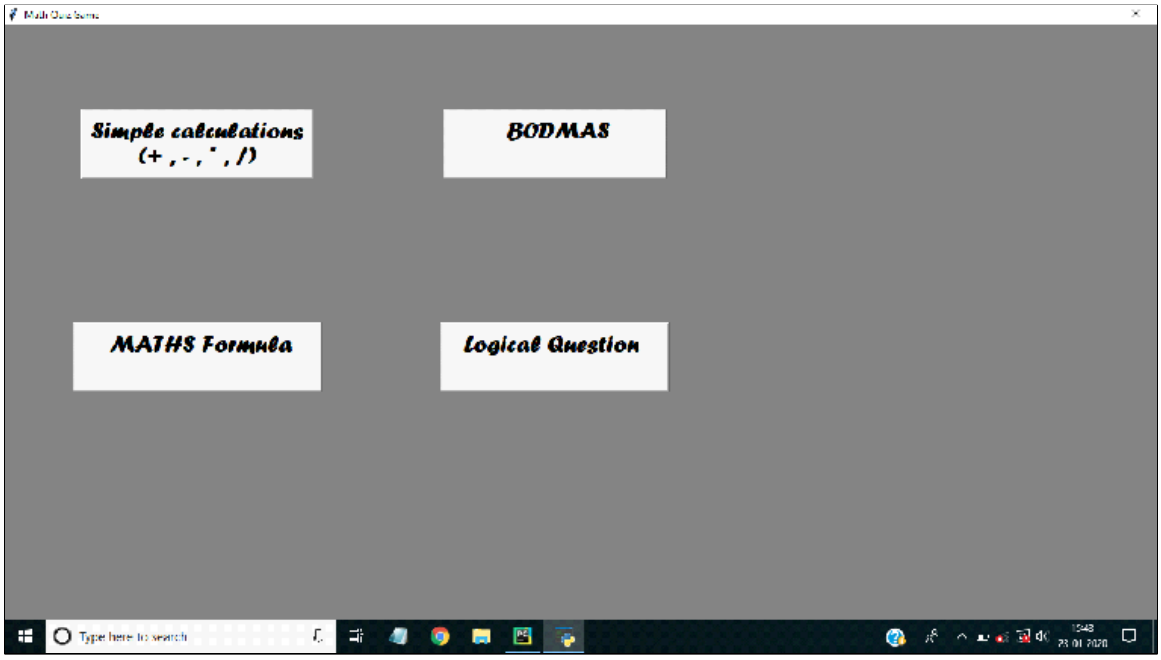
```
from tkinter import *
m1 = PanedWindow()
m1.pack(fill = BOTH, expand = 1)
```

```
left = Entry(m1, bd = 5)
m1.add(left)
m2 = PanedWindow(m1, orient = VERTICAL)
m1.add(m2)
top = Scale( m2, orient = HORIZONTAL)
m2.add(top)
mainloop()
```

RESULTS:

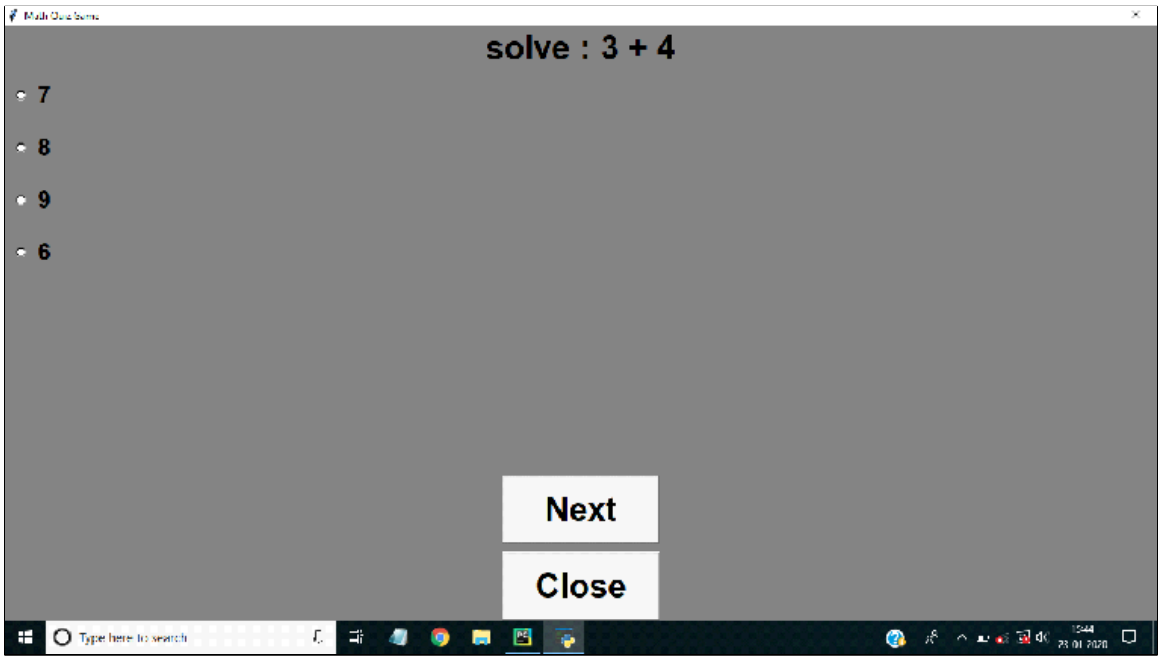
Splash Screen:

Main Activity:

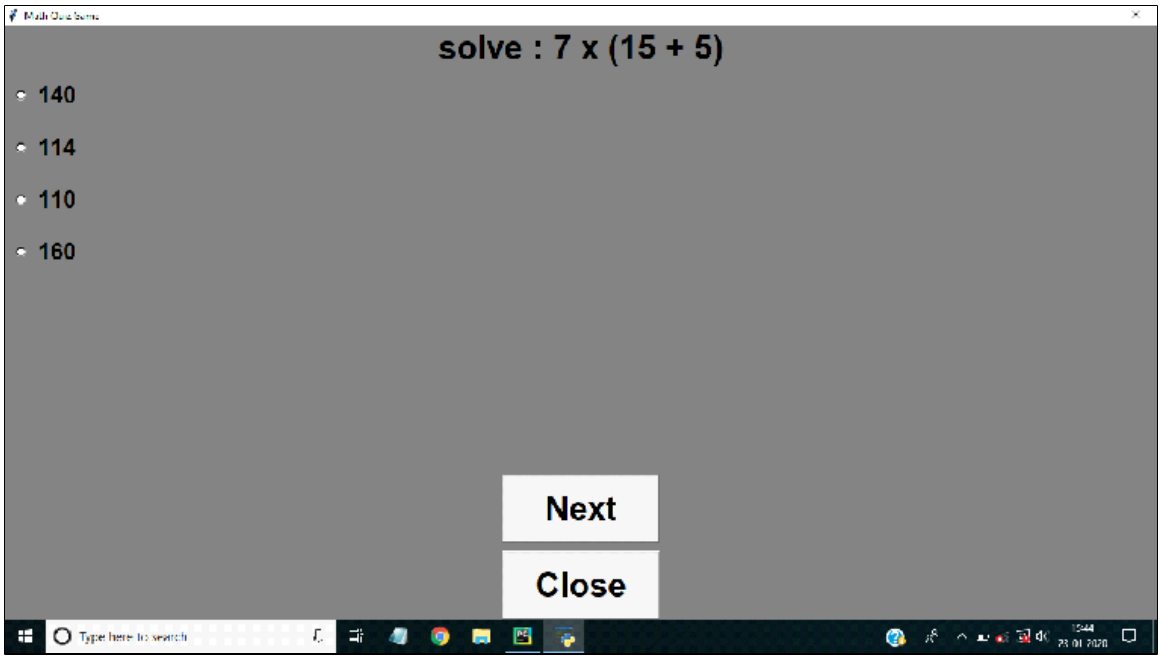


Quiz page(Game Start):

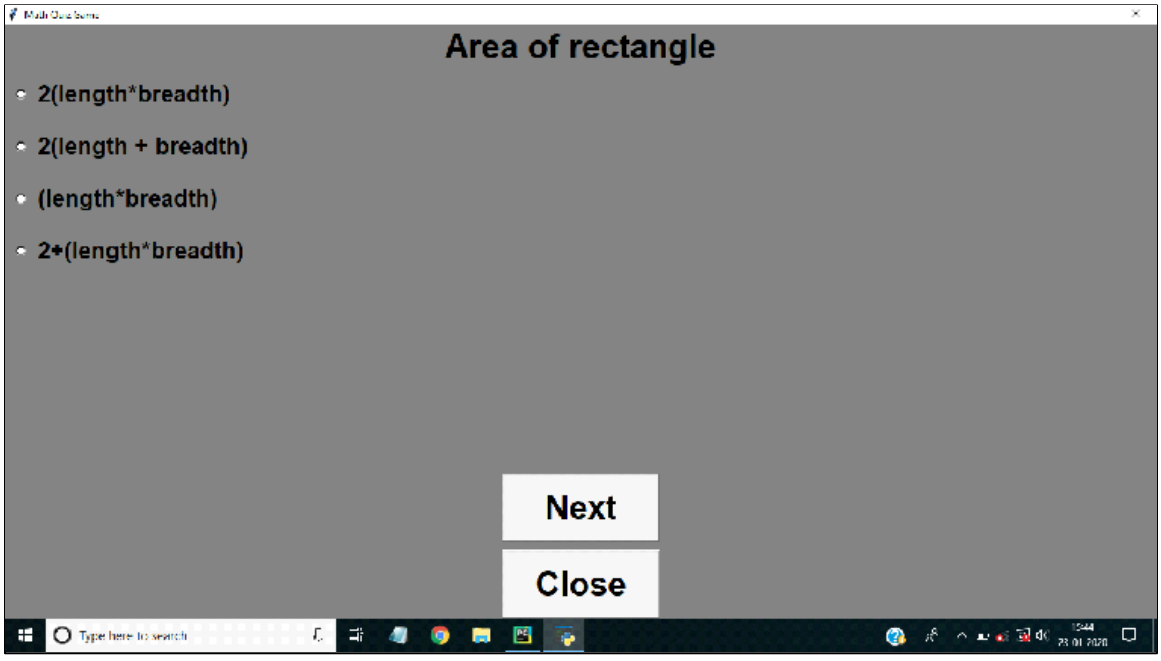
1.simple calculation



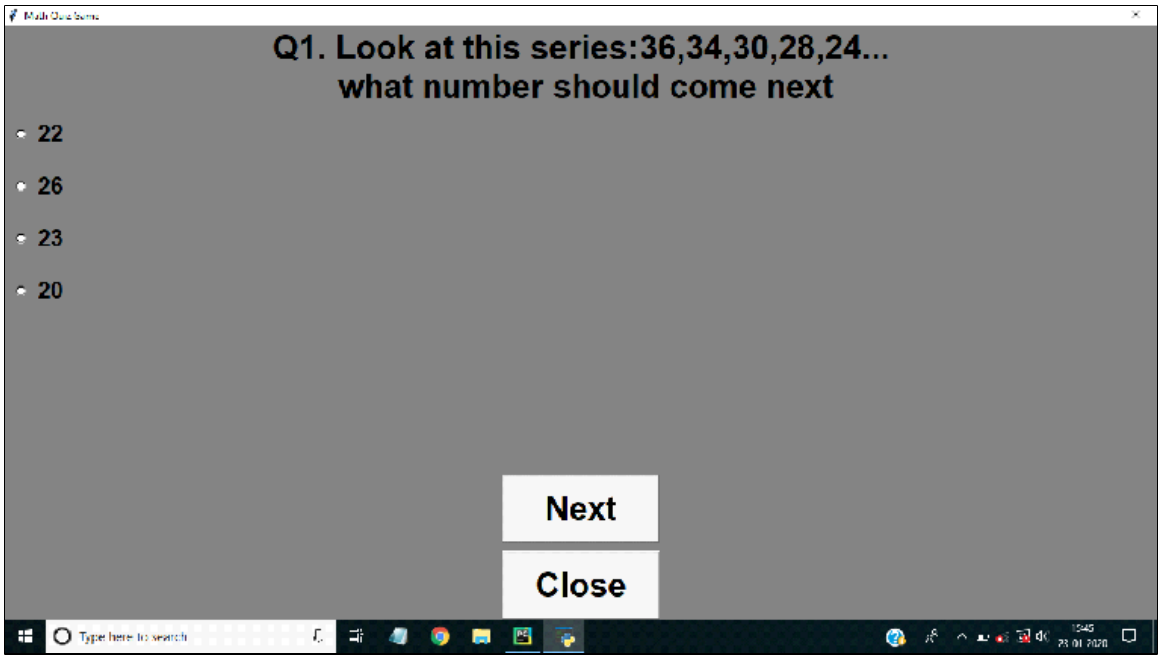
2.BODMAS Question



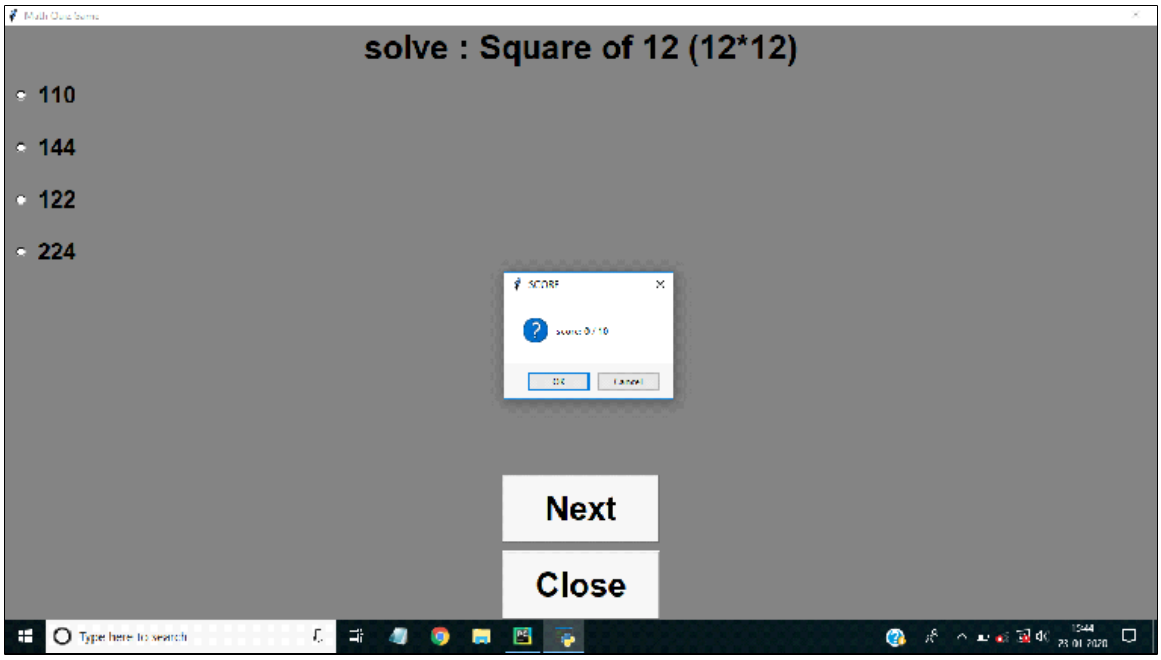
3.Math formula



4.Logical Question



SCORE PAGE:

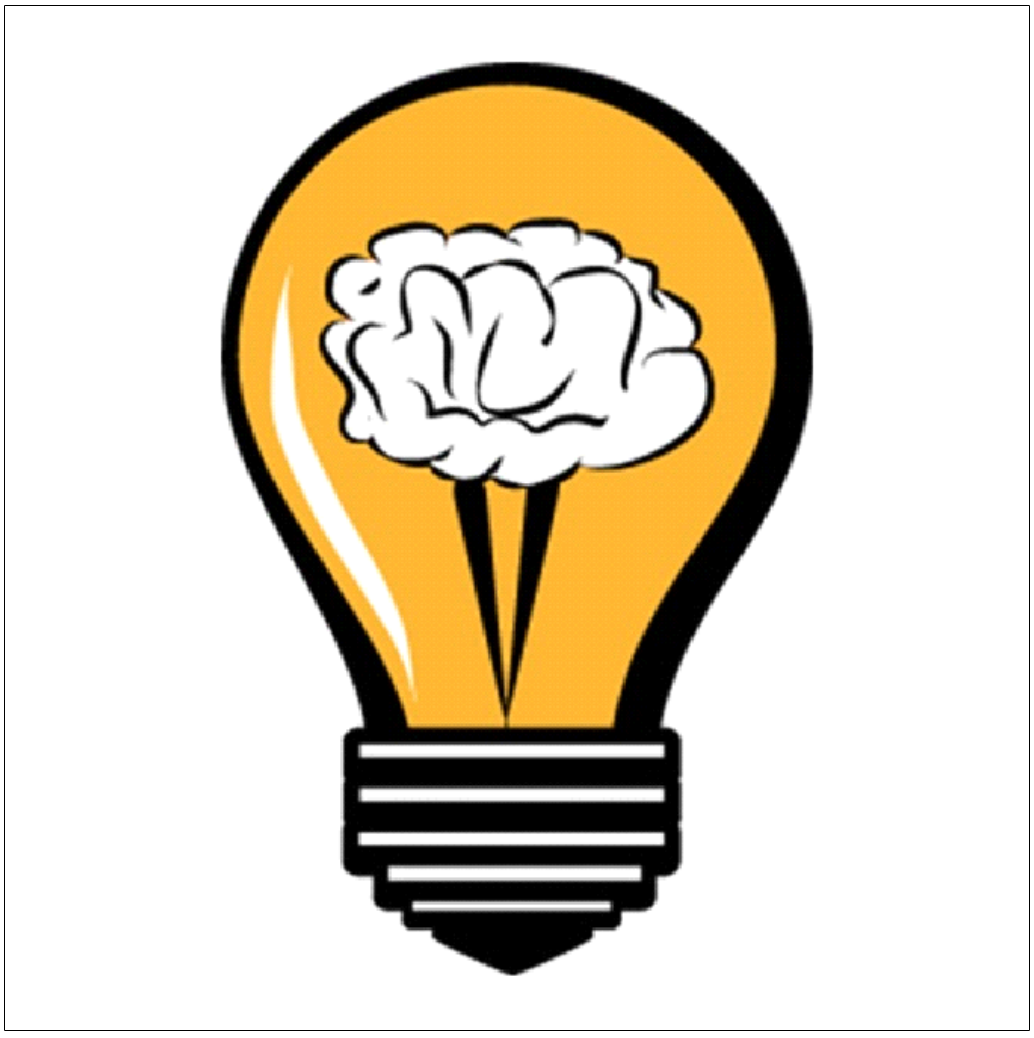


Conclusion:

Currently there are many medication reminder systems which are operable manually. Due to increased manual work, the available system becomes more time consuming. so in the given work, an attempt has been made to implement fully automatic medication reminder system.It eases the user's task of recalling whem to take the medicine by reminding them of the particular medicine at the correct time thereby reducing the much prevalent manual work.

Bibliography:

- www.wikipedia.com
- www.tutorialspoint.com
- www.geeksforgeeks.org



MATH QUIZ