H.K.E Society's
POOJYA DODDAPPA APPA COLLEGE OF ENGINEERING
KALABURAGI – 585102
(An Autonomous Institution, Affiliated to VTU Belagavi, and Approved By AICTE)

MINI PROJECT REPORT ON

"TYPING SPEED GAME"

Submitted to

POOJYA DODDAPPA APPA COLLEGE OF ENGINEERING
KALABURAGI

(An Autonomous Institution, affiliated to VTU Belagavi and Approved by AICTE)

In the partial fulfillment of the requirements for the Award of Degree of

BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE & ENGINEERING

SUBMITTED BY

RAVI KUMAR SINGH	3PD17CS077
SHARAN PATIL	3PD17CS090

UNDER THE GUIDANCE
DR. SHAILAJA

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
PDA COLLEGE OF ENGINEERING
AUTONOMOUS INSTITUITION
KALABURAGI
2019-2020

Hyderabad Karnataka education Society's
POOJYA DODDAPPA APPA COLLEGE OF ENGINEERING
KALABURAGI – 585102

(An autonomous Institution, affiliated to VTU Belagavi, and approved by AICTE)

Department of Computer Science and Engineering

CERTIFICATE

Certified that the project work entitled "TYPING SPEED GAME" is a bonafide work carried out by RAVI KUMAR SINGH (3PD17CS077) & SHARAN PATIL (3PD17CS090) in partial fulfilment of engineering in COMPUTER SCIENCE AND ENGINEERING of Poojya Doddappa Appa College of Engineering, Kalaburagi, an Autonomous Institution, affiliated to Visvesvaraya Technological University, Belagavi during the year 2019-20

Guide:
Dr. Shailaja

H.O.D
Dr. Survarna Nandyal

Examiners:

1.

2.

Declaration

I hereby declare that the work which is being represented in the mini project work entitled ***"TYPING SPEED GAME"*** *in COMPUTER SCIENCE AND ENGINEERING of Poojya Doddappa appa college of engineering kalaburagi, an autonomous institution, affiliated to Visvesvaraya Technology University, Belagavi, as a record is my own work carried out under the guidance of Dr. SHAILAJA*

Student's signature

Date:

RAVI KUMAR SINGH

Place: Kalaburagi

SHARAN PATIL

CONTENTS

Abstract

The world is moving forward with a rapid improvement in technology. The importance of computer skills is at all-time high. One such skill that in particular has become an ever-increasing attribute from staff or a prospective candidate is Touch Typing.

Most of the people working on a computer spend their time in typing some or the other thing. If the invaluable time that is spent typing on the computer is reduced than instantly, we get some more time available for other tasks during the day.

This report describes an implementation of a python-based game that will help you to check your typing speed and improve it with regular practice. This python project is build using pygame library and can detect typing speed of user with accuracy & words per minute. The pygame library is used for developing the graphical interface of the application.

1. Introduction

Touch Typing used to be a skill taught to people who wished to join the administration industry, now not only is it a must within this industry but every other industry can benefit from the skill.

As you are aware, most people who are working will come into contact with a computer during a standard working day. For some, this is a short time and others it can be all day. Now imagine if that invaluable time currently spent typing on your or your staff's computer was reduced or even halved! Instantly there is more time available for other tasks during the day, so if you have mounds of paperwork on your desk and you find yourself looking at it thinking 'this is going to take me all day', then developing the touch typing skill could be the answer.

The productivity of a business depends on how things are done faster. To complete your work faster it is important to develop typing skills. Typing helps you to work comfortably on the computer, it aids in communicating with colleagues and customers, creating documents, and finding new information.

Our project "Typing Speed Game" helps you in achieving the touch typing skills and also track the accuracy . Every time you look at the keyboard or make a mistake, you move your attention away from what you really want to do. But once you learn touch typing, You can easily type faster without giving much attention to the keyboard.

2. Software Requirements

- Python 3.x
- PyGame
- Operating System: Windows/Linux
- Any Python IDE

Python:

Python is a high level, general purpose programming language. It is multi-paradigm programming language. It provides support to Object-oriented programming and structured programming. Python also provides several features to support functional programming.

PyGame:

Pygame is a Python wrapper module for the SDL multimedia library. It contains python functions and classes that will allow you to use SDL's support for playing cd roms, audio and video output, and keyboard, mouse and joystick input. We can install the pygame module with the pip tool which is used by python to install packages. The pygame module is used for the following:

- Draw items on the screen
- Handle user input
- Implement user event loops

Installation of PyGame:

- *pip install pygame*

3. Working Principle

In order to understand the working of the project, let us first understand the file structure of the project.

- Background.jpg – A background image we will use in our program.
- Icon.png – An icon image that we will use as a reset button.
- Sentences.txt – This text file will contain a list of sentences separated by a new line.
- Speed typing.py – The main program file that contains all the code.
- Typing-speed-open.png – The image to display when starting game.

The project follows an Object-Oriented approach which allows us to break down functionalities into different methods and classes. This part of report contains the explanation of each part of the code.

This project uses pygame module to implement the graphical user interface and other functionalities like keeping track of keyboard and mouse events. Therefore, first of all we need to import the pygame library along with some other built-in modules which are sys, time and random.

Game Class:

The game class includes the functions that are required in the project. These functions are responsible for starting the game, resetting the game and some of the helper functions.

Functions used in Game Class:

This first method is the constructor method for our class where we have defined the variables we use in out project. The constructor initializes the width and height of the window, variables that are needed for calculation. We have also initialized the pygame and loaded the images from file.

1. draw_text()

The draw_text() method of Game class is a helper function that will draw the text on the screen. The argument it takes is the screen, the message we want to draw, the y coordinate of the screen to position our text, the size of the font and colour of the font. We will draw everything in the centre of the screen. After drawing anything on the screen, pygame requires you to update the screen.

2. get_sentence()

Next is the get_sentence() method. The get_sentence() method will open up the file "sentence.txt" and return a random sentence from the list. We split the whole string with a newline character.

3. show_results()

The show_results() method is where we calculate the typing speed of the user. The time starts when the user clicks on the input box and when the user hits return key "Enter" then we perform the difference and calculate time in seconds.

To calculate accuracy, we did a little bit of math. We counted the correct typed characters by comparing input text with the display text which the user had to type.

The formula for accuracy is:

(correct characters)*100/ (total characters in sentence)

The WPM is the words per minute. A typical word consists of around 5 characters, so we calculate the words per minute by dividing the total number of words with five and then the result is again divided that with the total time it took in minutes. Since our total time was in seconds, we had to convert it into minutes by dividing total time with 60.

4. Run()

The run() method is the method that will start the game. This is also the main method of our class that will handle all the events. We call the reset_game() method at the starting of this method in order to reset the variables to initial value. Next, we run an infinite loop which will capture all the mouse and keyboard events. Then, we draw the heading and the input box on the screen.

We then use another loop that will look for the mouse and keyboard events. When the mouse button is pressed, we check the position of the mouse if it is on the input box then we start the time and set the active to True. If it is on the reset button, then we reset the game.

When the active is True and typing has not ended then we look for keyboard events. If the user presses any key then we need to update the message on our input box. The enter key will end typing and we will calculate the scores to display it. Another event of a backspace is used to trim the input text by removing the last character.

We have placed a reset button at the bottom of our window. When the user clicks on this button the rest_game() method is called. The reset_game() method resets all variables so that we can start testing our typing speed again. We also select a random sentence by calling the get_sentence() method. In the end, we have closed the class definition and created the object of Game class to run the program.

4. Code

```
import pygame
from pygame.locals import *
import sys
import time
import random
#         750
x   500   class
Game:


        def __init__(self):
                self.w=750
                self.h=500
                self.reset=True
                self.active = False
                self.input_text=''
                self.word = ''
                self.time_start = 0
                self.total_time = 0
                self.accuracy = '0%'
                self.results = 'Time:0 Accuracy:0 % Wpm:0 '
                self.wpm = 0
                self.end = False
                self.HEAD_C = (255,213,102)
                self.TEXT_C = (240,240,240)
```

```
        self.RESULT_C = (255,70,70)


        pygame.init()
        self.open_img = pygame.image.load('type-speed-open.png')
        self.open_img = pygame.transform.scale(self.open_img, (self.w,self.h))

        self.bg = pygame.image.load('background.jpg')
        self.bg = pygame.transform.scale(self.bg, (500,750))
        self.screen = pygame.display.set_mode((self.w,self.h))
        pygame.display.set_caption('Type Speed test')


def draw_text(self, screen, msg, y ,fsize, color):
        font = pygame.font.Font(None, fsize)
        text = font.render(msg, 1,color)
        text_rect = text.get_rect(center=(self.w/2, y))
        screen.blit(text, text_rect)
        pygame.display.update()
def get_sentence(self):
        f          = open('sentences.txt').read()
        sentences = f.split('\n') sentence =
        random.choice(sentences) return sentence
def show_results(self, screen):
        if(not self.end):
                #Calculate time
                self.total_time = time.time() - self.time_start
                #Calculate accuracy
                count = 0
```

```
                for i,c in enumerate(self.word):
                    try:
                        if self.input_text[i] == c:
                            count += 1
                    except:
                        pass
                self.accuracy = count/len(self.word)*100
                #Calculate words per minute
                self.wpm = len(self.input_text)*60/(5*self.total_time)
                self.end = True
                print(self.total_time)
                self.results = 'Time:'+str(round(self.total_time)) +" secs Accur acy:"+
str(round(self.accuracy)) + "%" + ' Wpm: ' + str(round(self.wpm))
                # draw icon image
                self.time_img = pygame.image.load('icon.png')
                self.time_img = pygame.transform.scale(self.time_img, (150,150))
                #screen.blit(self.time_img, (80,320))
                screen.blit(self.time_img, (self.w/2-75,self.h-140))
                self.draw_text(screen,"Reset", self.h - 70, 26, (100,100,100))
                print(self.results)
                pygame.display.update()
    def run(self):
        self.reset_game()

        self.running=True
        while(self.running):
            clock = pygame.time.Clock()
```

```
                self.screen.fill((0,0,0), (50,250,650,50))
                pygame.draw.rect(self.screen,self.HEAD_C, (50,250,650,50), 2)
                # update the text of user input
                self.draw_text(self.screen, self.input_text, 274, 26,(250,250,250)
)
                pygame.display.update()
                for event in pygame.event.get():
                    if event.type == QUIT:
                        self.running = False
                        sys.exit()
                    elif event.type == pygame.MOUSEBUTTONUP:
                        x,y = pygame.mouse.get_pos()
                        # position of input box
                        if(x>=50 and x<=650 and y>=250 and y<=300):
                            self.active = True
                            self.input_text = ''
                            self.time_start = time.time()
                         # position of reset box
                        if(x>=310 and x<=510 and y>=390 and self.end):
                            self.reset_game()
                             x,y = pygame.mouse.get_pos()

                    elif event.type == pygame.KEYDOWN:
                        if self.active and not self.end:
                            if event.key == pygame.K_RETURN:
                                print(self.input_text)
```

```
                                    self.show_results(self.screen)
                                    print(self.results)
                                    self.draw_text(self.screen, self.results,350, 28,
self.RESULT_C)
                                    self.end = True
                                elif event.key == pygame.K_BACKSPACE:
                                    self.input_text = self.input_text[:-1]
                                else:
                                    try:
                                        self.input_text += event.unicode
                                    except:
                                        pass
                    pygame.display.update()

            clock.tick(60)
        def reset_game(self):
            self.screen.blit(self.open_img, (0,0))
            pygame.display.update()
            time.sleep(1)
            self.reset=False
            self.end = False
            self.input_text=''
            self.word = ''
            self.time_start = 0
            self.total_time = 0
            self.wpm = 0
```

```
            # Get random sentence
            self.word = self.get_sentence()
            if (not self.word): self.reset_game()
            #drawing heading
            self.screen.fill((0,0,0))
            self.screen.blit(self.bg,(0,0))
            msg = "Typing Speed Test"
            self.draw_text(self.screen, msg,80, 80,self.HEAD_C)
            # draw the rectangle for input box
                pygame.draw.rect(self.screen,(255,192,25), (50,250,650,50), 2)
            # draw the sentence string
            self.draw_text(self.screen, self.word,200, 28,self.TEXT_C)
            pygame.display.update()


Game().run()
```

5. Applications

This project provides an easy to use and friendly user interface which allows the users to use the app with ease. Users who learn Touch typing can play this game to check their progress.

Typing speed can be improved by daily practiced and this project lets the user take up the challenge and check their progress with this game.

The major benefits of speed typing are:

- Save time:

  It would help you to write your blogs, letters, answers, chatting etc. faster without wasting time.

- Impression:

  It will impress other people who don't know this skill. They may say like, Wow! how can you do that and within mind they may be thinking if they could also type like you. Imagine!

- Pleasure:

  It is a pleasure to just focus on thinking and the typing is done automatically unlike others who don't know this skill.

- Job:

  Many jobs both in govt. and private sector ask for good typing speed.

- Lifetime with you:

  Like bicycle, once you learn it, will be always with you.

6. Result and Discussion

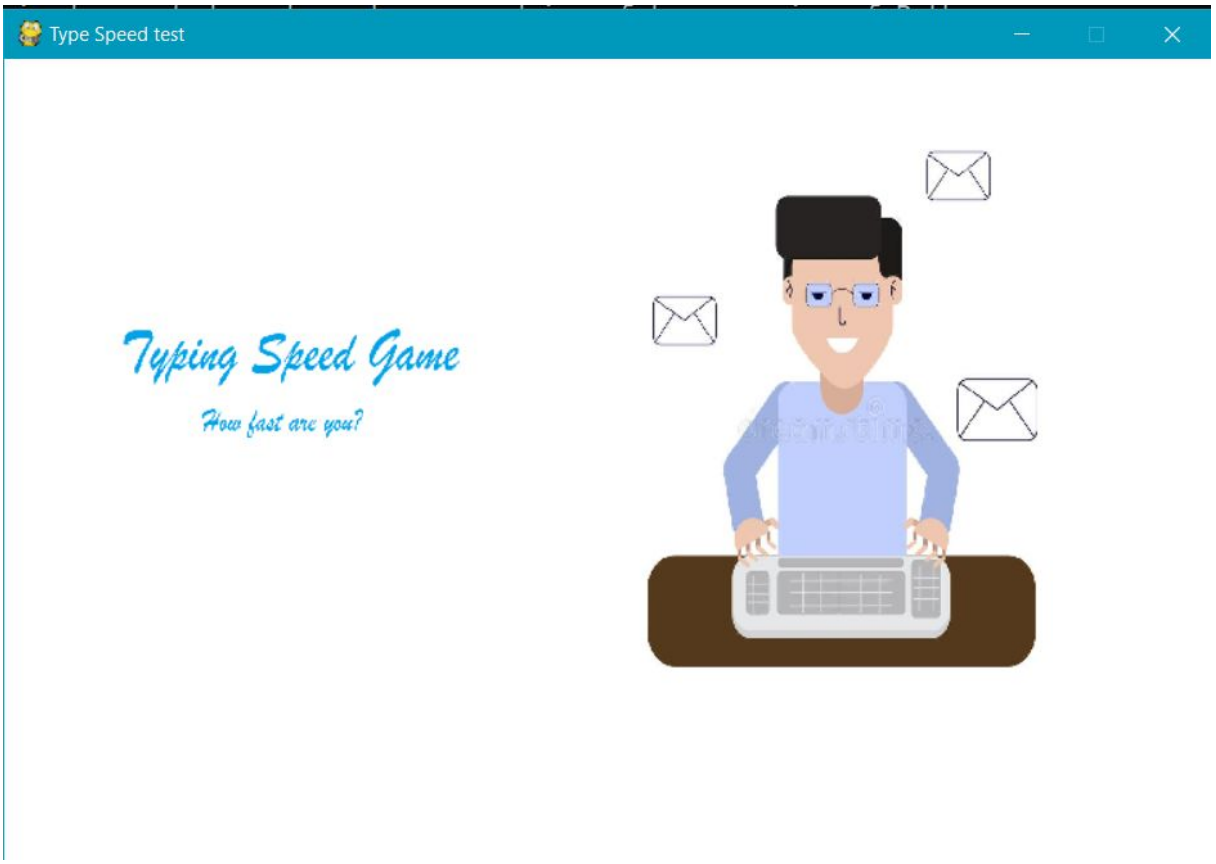

Figure 1: Opening Screen

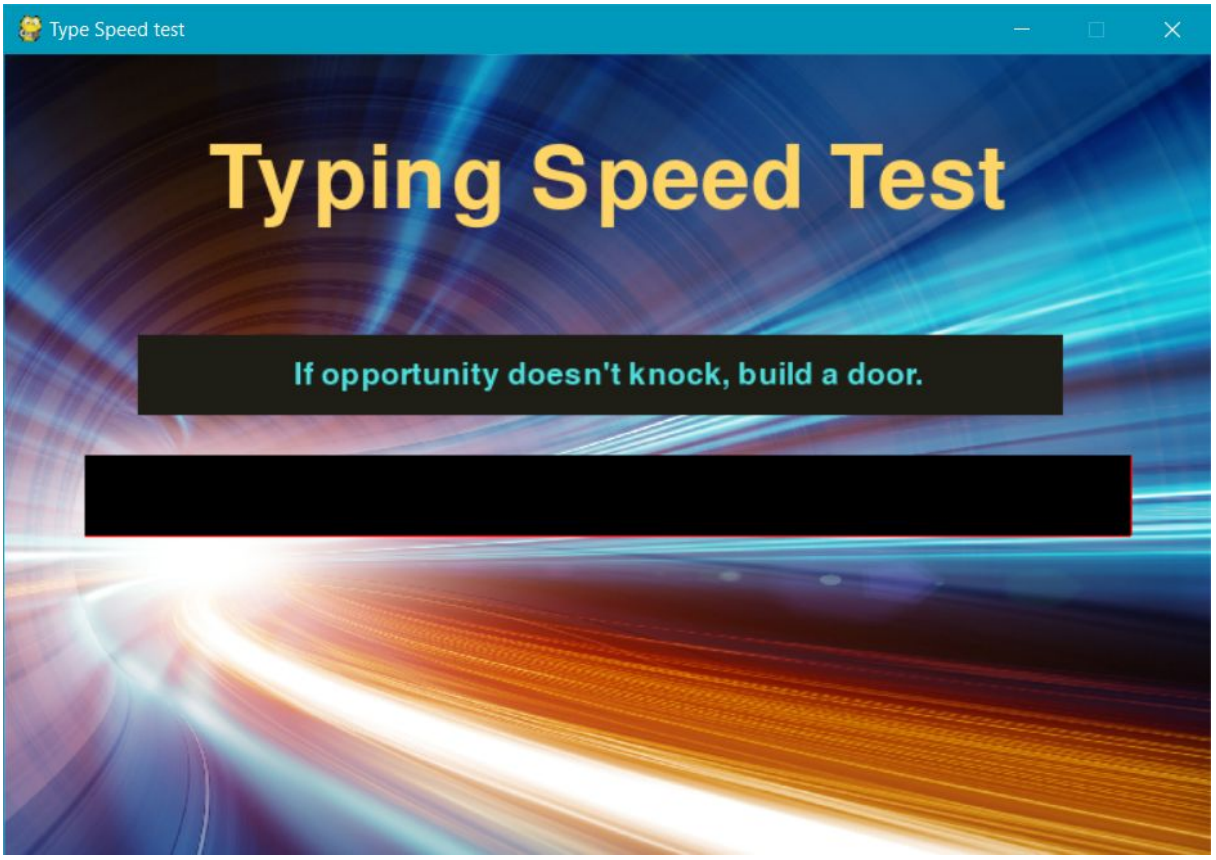

Figure 2: Game Screen

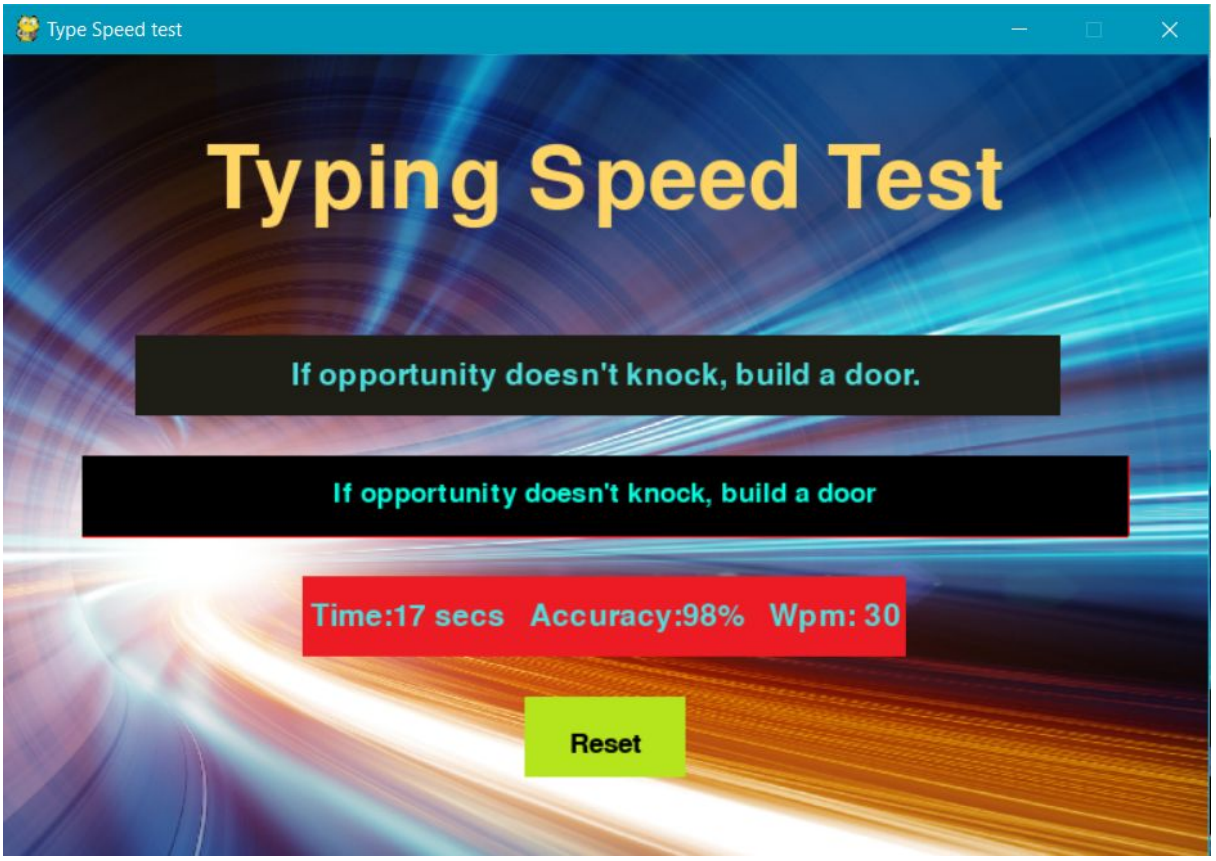

Figure 3: Result Screen

Figure 1: Opening Screen

This is the screen which appears when the game is opened. We have used a background image for this loading screen. It stays for 1 second and then the main screen appears.

Figure 2: Game Screen

This is the main screen of the game. A sentence which must be typed by the user is displayed on the screen along with a header which display the Title of our game and a input box. Below the input box we have the reset button which is used to restart the game.

Figure 3: Result Screen

Once the used types the sentence and hits enter key the speed, accuracy and time taken is calculated and displayed on the screen above the reset button.

The game has a very simple user interface which makes is easy for the naïve users to interact with the game and practice typing . Touch typing is an important skill to learn which stay with you forever.

7. Conclusion

In this paper we have presented a Python game which will help the users to improve their typing speed by playing the game. The game will help naïve users to learn the key positions on the keyboard. Anyone can learn touch typing and improve their typing speed by regular practice. This game helps user practice the typing and gives results at the end which will be useful to track the progress made by the user. Touch typing may seem like it's not worth the time, especially if you're already confident in your hunt and peck skills. It is, however, one of the most valuable skills you can learn. It may take a little more time than you would like, especially if you have a life time of bad habits to relearn, but in the end, a small investment of time will pay off in more ways than you can imagine.