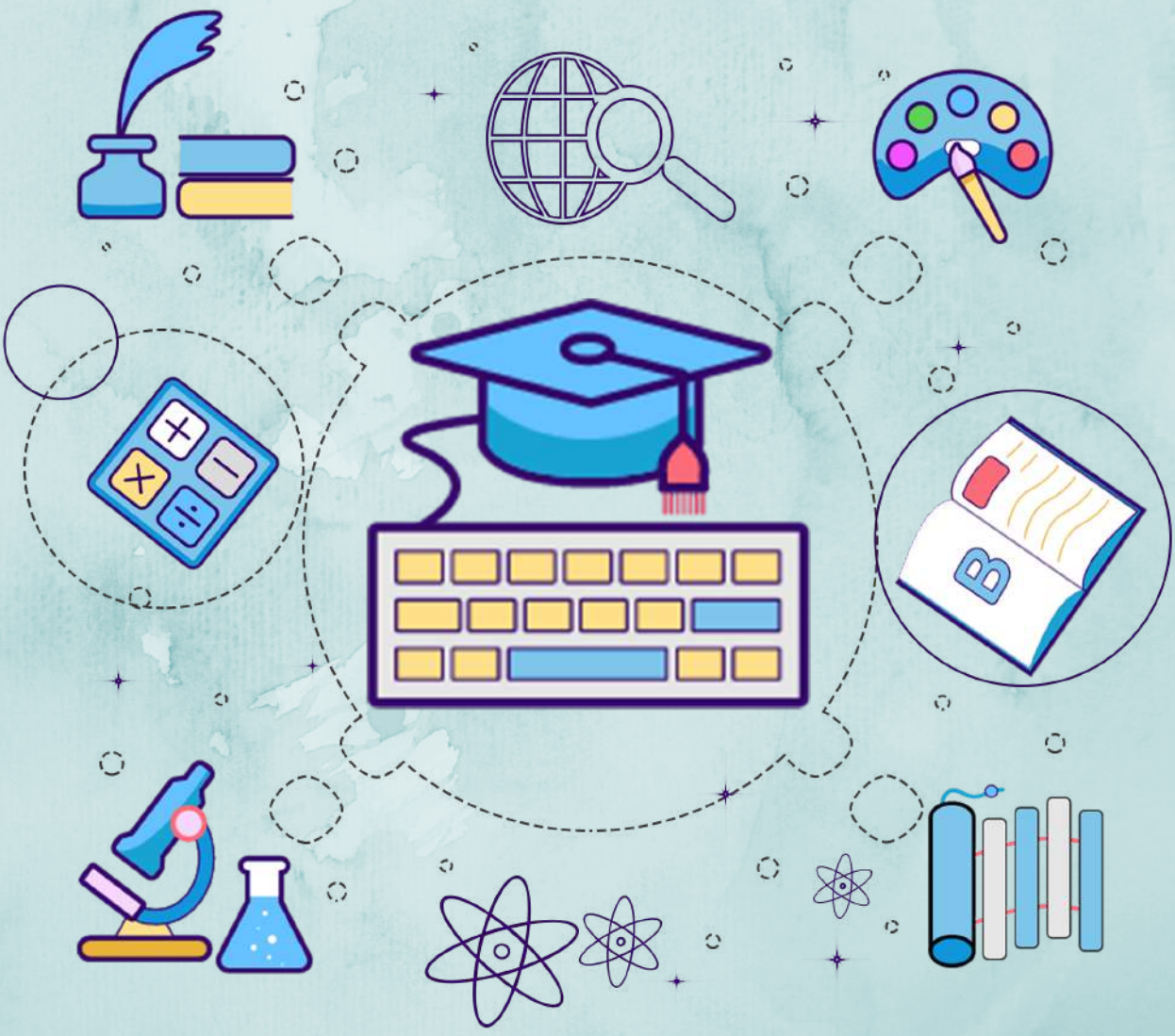
APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

KeralaNotes



SYLLABUS | STUDY MATERIALS | TEXTBOOK

PDF | SOLVED QUESTION PAPERS

www.keralanotes.com

KeralaNotes

KTU STUDY MATERIALS

PROGRAMMING IN PYTHON

CST 362

Module 3

Related Link :

- KTU S6 CSE NOTES 2019 SCHEME
- KTU S6 SYLLABUS CSE COMPUTER SCIENCE
- KTU PREVIOUS QUESTION BANK S6 CSE SOLVED
- KTU CSE TEXTBOOKS S6 B.TECH PDF DOWNLOAD
- KTU S6 CSE NOTES | SYLLABUS | QBANK | TEXTBOOKS DOWNLOAD

TUTORIAL QUESTIONS

MODULE-III

1. Differentiate between terminal based and GUI based programming in Python?

ANS:

GUI	CLI
A type of user interface that allows users to interact with electronic devices through graphical icons and visual indicators	An interface for the user to issue commands in the form of successive lines of text or command lines to perform the tasks
Graphical User Interface	Command Line Interface
Even a beginner can easily handle	User should have good knowledge of commands
Requires more memory as it contains a lot of graphical components	Does not require more memory
Slower	Fast
There are customizable options to change the appearance	It is not possible to change the apperance
More flexible	Not much flexible

2. Write a program to draw a hexagon using turtle. (university question)

Ans:

```
from turtle import Turtle
t=Turtle()
def hexagon(t, length):
  for count in range(6):
    t.forward(length)
    t.left(60)
hexagon(t,100)
```

3. Write a program to draw a pentagon using turtle.

Ans:

```
from turtle import Turtle
t=Turtle()
def pentagon(t, length):
  for count in range(5):
    t.forward(length)
    t.left(72)
pentagon(t,100)
```

4. Write a program to draw a star using turtle.

Ans:

```
from turtle import Turtle
t=Turtle()
def star(t, length):
  for count in range(5):
    t.forward(length)
```

```
    t.left(144)
star(t,100)
```

5. Write a program to draw a circle using turtle.

Ans:

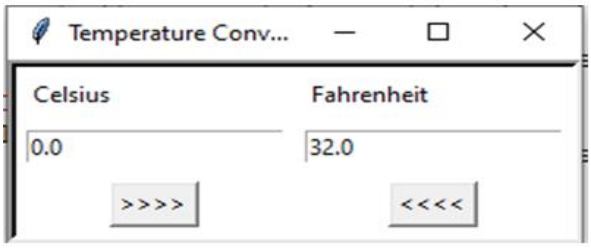
```
from turtle import Turtle
t=Turtle()
r=100
t.circle(r)
```

6. Write a GUI based program that allows the user to convert temperature values between degrees Fahrenheit and degrees Celsius. The interface should have labeled entry fields for these two values. These components should arrange in a grid where the labels occupy the first row and the corresponding field occupy the second row. At start up the Fahrenheit field should contain 32 and the Celsius field contain 0.0.The third row in the window contain two command buttons ,labeled >>>> and <<<<.When the user presses the first button, the program should use the data in the Fahrenheit field to compute the Celsius value, which should then be output to the Celsius field. The second button should perform the inverse function? (university question)

Ans:

```
from breezypythongui import EasyFrame
class NumberFieldDemo(EasyFrame):
    def __init__(self):
        EasyFrame.__init__(self, title = "Temperature Conversion")
        self.addLabel(text = "Celsius", row = 0, column = 0)
        self.celsiusField = self.addIntegerField(value = 0, row = 0,column=1,
```

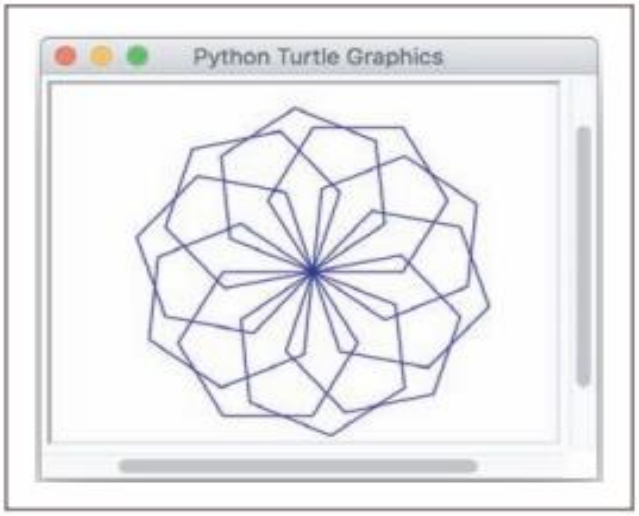
```
 width = 10)
 self.far=self.addLabel(text = "Fahrenheit",row = 1, column = 0)
 self.fahrenField = self.addFloatField(value = 0.0, row =1,column = 1,
 width = 8, precision = 2)
# The command button
 self.addButton(text = ">>>>", row = 2, column = 0, columnspan = 2,
 command = self.computefahr)
 self.addButton(text = "<<<<", row = 2, column = 1,columnspan = 2,
 command = self.computecelsius)
# The event handling method for the button
def computefahr(self):
    number = self.celsiusField.getNumber()
    result = number * 1.8 + 32
    self.fahrenField.setNumber(result)
 def computecelsius(self):
   number = self.fahrenField.getNumber()
   result = ((number-32)*5)/9
   self.celsiusField.setNumber(result)
NumberFieldDemo().mainloop()
```



7. Write a program to draw a radial pattern with 10 hexagons.

Ans:

```
from turtle import Turtle
t=Turtle()
def radialHexagons(t, n, length):
    for count in range(n):
        hexagon(t, length)
        t.left(360 / n)
def hexagon(t, length):
    for count in range(6):
        t.forward(length)
        t.left(60)
radialHexagons(t,10,100)
```



8. Write a GUI program to convert an input string to uppercase and displays the result.

Ans:

```
from breezypythongui import EasyFrame
class TextFieldDemo(EasyFrame):
        def __init__(self):
                EasyFrame.__init__(self, title = "Text Field Demo")
                self.addLabel(text = "Input", row = 0, column = 0)
                self.inputField = self.addTextField(text = "",row=0,column = 1)
```

```
            self.addLabel(text = "Output", row = 1, column = 0)
            self.outputField=self.addTextField(text="",row=1,column=1)
            self.addButton(text = "Convert", row=2,column= 0,columnspan
= 2, command = self.convert)
        # The event handling method for the button
        def convert(self):
            text = self.inputField.getText()
            result = text.upper()
            self.outputField.setText(result)
TextFieldDemo().mainloop()
```

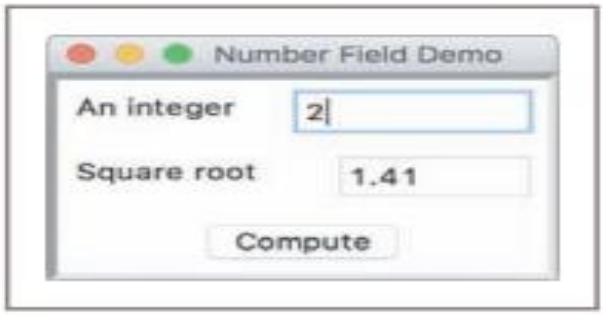
9. Write GUI program to compute and displays the square root of an input number.

Ans:

```
from breezypythongui import EasyFrame
import math
class NumberFieldDemo(EasyFrame):
    def __init__(self):
```

```
        EasyFrame.__init__(self, title = "Number Field Demo")
        self.addLabel(text = "An integer",row = 0, column = 0)
        self.inputField = self.addIntegerField(value = 0,row = 0,column = 1,
        width = 10)
        self.addLabel(text = "Square root",row = 1, column = 0)
        self.outputField = self.addFloatField(value = 0.0,row = 1,
        column = 1,width = 8,precision = 2)

        self.addButton(text = "Compute", row = 2, column = 0,
        columnspan = 2, command = self.computeSqrt)
        # The event handling method for the button
    def computeSqrt(self):
        number = self.inputField.getNumber()
        result = math.sqrt(number)
        self.outputField.setNumber(result)
NumberFieldDemo().mainloop()
```



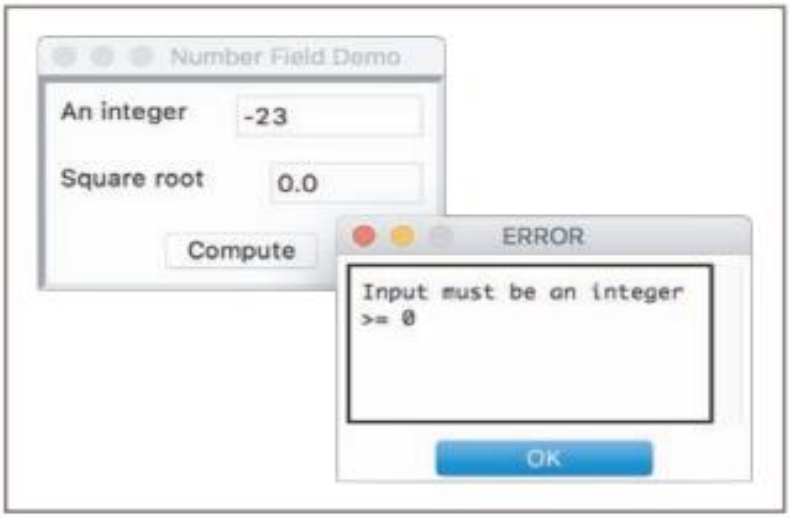
10. Write a GUI program to inputs the integer, computes the square root, and outputs the result and handles input errors by displaying a message box.

Ans:

```
from breezypythongui import EasyFrame
import math
class NumberFieldDemo(EasyFrame):
    def __init__(self):
        EasyFrame.__init__(self, title = "Number Field Demo")
        self.addLabel(text = "An integer",row = 0, column = 0)
        self.inputField = self.addIntegerField(value = 0,row = 0,column = 1,
        width = 10)
        self.addLabel(text = "Square root",row = 1, column = 0)
        self.outputField = self.addFloatField(value = 0.0,row = 1,
        column = 1,width = 8,precision = 2)

        self.addButton(text = "Compute", row = 2, column = 0,
        columnspan = 2, command = self.computeSqrt)
        # The event handling method for the button
    def computeSqrt(self):
        try:
            number = self.inputField.getNumber()
            result = math.sqrt(number)
            self.outputField.setNumber(result)
```

```
        except ValueError:
            self.messageBox(title = "ERROR", message = "Input must
            be   an integer >= 0")
NumberFieldDemo().mainloop()
```



11. Write a GUI-based program that plays a guess-the-number game in which the user guesses a number between 1 and 100 and the computer provides the responses. The window should display the user's guesses with a label by saying, "Too large, try again," or "Too small, try again." When the user finally guesses the correct number, the program congratulates him and tells him the total number of guesses.

```
import random
from breezypythongui import EasyFrame
class GuessingGame(EasyFrame):

    def __init__(self):
```

```
    """Sets up the window, widgets, and data."""
    EasyFrame.__init__(self, title = "Guessing Game")
    # Initialize the instance variables for the data
    self.myNumber = random.randint(1, 100)
    self.count = 0
    # Create and add widgets to the window
    greeting = "Guess a number between 1 and 100."
    self.hintLabel = self.addLabel(text = greeting,row = 0, column = 0,
   sticky   ="NSEW",columnspan = 2)
    self.addLabel(text = "Your guess", row = 1, column = 0)
    self.guessField = self.addIntegerField(0, row = 1, column = 1)
    # Buttons have no command attributes yet
    self.nextButton = self.addButton(text = "Next", row = 2,
     column = 0,command=self.nextGuess)
    self.newButton = self.addButton(text = "New game",row = 2,
     column = 1,command=self.newGame)
def nextGuess(self):
    """Processes the user's next guess."""
    self.count += 1
    guess = self.guessField.getNumber()
    if guess == self.myNumber:
        self.hintLabel["text"] = "You've guessed it in " +str(self.count)+"attempts!"
```

```
            self.nextButton["state"] = "disabled"
        elif guess < self.myNumber:
            self.hintLabel["text"] = "Sorry, too small!"
        else:
            self.hintLabel["text"] = "Sorry, too large!"
    def newGame(self):
        """Resets the data and GUI to their original states."""
        self.myNumber = random.randint(1, 100)
        self.count = 0
        greeting = "Guess a number between 1 and 100."
        self.hintLabel["text"] = greeting
        self.guessField.setNumber(0)
        self.nextButton["state"] = "normal"
GuessingGame().mainloop()
```

12. Write a GUI-based program that plays a guess-the-number game in which the computer guesses a number between 1 and 100 and the user provides the responses. The window should display the computer's guesses with a label. The user enters a hint in response, by selecting one of a set of command buttons labeled Too small, Too large, and Correct. When the game is over, you should

disable these buttons and wait for the user to click New game, as before. (university question)

Ans:

```
import random
from breezypythongui import EasyFrame
class GuessingGame(EasyFrame):


    def __init__(self):
        EasyFrame.__init__(self, title = "Guessing Game")
        self.myNumber = random.randint(1, 100)
        self.count = 0
        self.hintLabel = self.addLabel(text = "",row = 0, column = 0,
        sticky = "NSEW",columnspan = 2)
        self.addLabel(text = "Computer guess", row = 1, column = 0)
        self.guessField = self.addIntegerField(0, row = 1, column = 1)
        self.guessField.setNumber(self.myNumber)


        self.smallButton = self.addButton(text = "Too small", row = 2,
          column =0,command=self.Guess)
        self.largeButton = self.addButton(text = "Too large",row = 2,
                column = 1,command=self.Guess)
        self.correctButton = self.addButton(text = "Correct", row = 2,
                    column = 2,command=self.Guess)
```

```
    self.newButton = self.addButton(text = "New Game", row = 2,
        column = 3,command=self.newGame)
def Guess(self):
    self.count += 1
    Response=30
    if Response == self.myNumber:
        self.hintLabel["text"] = "You've guessed it in " +str(self.count) +
         "  attempts!"
        self.smallButton["state"] = "disabled"
        self.largeButton["state"] = "disabled"
        self.correctButton["state"] = "disabled"
    elif Response < self.myNumber:
        self.myNumber = random.randint(1, 100)
        self.guessField.setNumber(self.myNumber)
    else:
        self.myNumber = random.randint(1, 100)
        self.guessField.setNumber(self.myNumber)
def newGame(self):
    """Resets the data and GUI to their original states."""
    self.myNumber = random.randint(1, 100)
    self.count = 0
    greeting = ""
```

```
        self.hintLabel["text"] = greeting
        self.guessField.setNumber(self.myNumber)
        self.smallButton["state"] = "normal"
        self.largeButton["state"] = "normal"
        self.correctButton["state"] = "normal"
GuessingGame().mainloop()
```