```
""" helloIO.py
    illustrate form IO
    (still procedural)
    works, but bad design:
    no main function
"""

from Tkinter import *

app = Tk()
lblOutput = Label(app, text = "type your name")
lblOutput.grid()

txtInput = Entry(app)
txtInput.grid()

btnClickMe = Button(app, text = "Click Me")
btnClickMe.grid()

def sayHi():
  name = txtInput.get()
  lblOutput["text"] = "Hi there, %s!" % name

btnClickMe["command"] = sayHi
# note no parens after sayHi
# treating function as a variable

app.mainloop()
```

```
""" helloBroken.py
    illustrate form IO
    Using functions causes problems
    sayHi doesn't know what lblOutput is!
    This program will not work!
"""

from Tkinter import *

def sayHi():
  name = txtInput.get()
  lblOutput["text"] = "Hi there, %s!" % name

def main():
  app = Tk()
  lblOutput = Label(app, text = "type your name")
  lblOutput.grid()

  txtInput = Entry(app)
  txtInput.grid()

  btnClickMe = Button(app, text = "Click Me")
  btnClickMe.grid()

  btnClickMe["command"] = sayHi

  app.mainloop()

if __name__ == "__main__":
  main()
```

```
""" gradeCalc.py
    calculate grades with a GUI
"""

from Tkinter import *

class App(Tk):
  def __init__(self):
    Tk.__init__(self)

    self.headerFont = ("Helvetica", "16", "bold italic")

    self.title("Grade Calculator")
    self.addLabs()
    self.addExams()
    self.addProj()
    self.addOutput()

  def addLabs(self):
    """ add lab elements """
    # Labels can be anonymous (no variable created)
    # just 'chain' the grid command
    # use columnspan to span multiple columns
    # use font to adjust the font size
    Label(self, text = "Lab Assignments",
          font = self.headerFont).grid(columnspan = 2)

    #explicitly set row and column
    Label(self, text = "lab 1").grid(row = 1, column = 0)
    self.txtLab1 = Entry(self)
    self.txtLab1.grid(row = 1, column = 1)
    # Entry's insert() method adds default text.
    # first parameter is position of cursor
    # second parameter is value to add
    self.txtLab1.insert(0, "100")

    Label(self, text = "lab 2").grid(row = 2, column = 0)
    self.txtLab2 = Entry(self)
    self.txtLab2.grid(row = 2, column = 1)
    self.txtLab2.insert(0, "100")

    Label(self, text = "lab 3").grid(row = 3, column = 0)
    self.txtLab3 = Entry(self)
    self.txtLab3.grid(row = 3, column = 1)
    self.txtLab3.insert(0, "100")

    Label(self, text = "lab 4").grid(row = 4, column = 0)
    self.txtLab4 = Entry(self)
    self.txtLab4.grid(row = 4, column = 1)
    self.txtLab4.insert(0, "100")

  def addExams(self):
    """ add exam elements """
```

```
    Label(self, text = "Exams",
          font = self.headerFont).grid(row = 5, columnspan = 2)

    Label(self, text = "Midterm Exam").grid(row = 6, column = 0)
    self.txtMT = Entry(self)
    self.txtMT.grid(row = 6, column = 1)
    self.txtMT.insert(0, "100")

    Label(self, text = "Final Exam").grid(row = 7, column = 0)
    self.txtFE = Entry(self)
    self.txtFE.grid(row = 7, column = 1)
    self.txtFE.insert(0, "100")

  def addProj(self):
    """ add project """
    Label(self, text = "Final Project",
          font = self.headerFont).grid(row = 8, columnspan = 2)

    Label(self, text = "Project").grid(row = 9, column = 0)
    self.txtFP = Entry(self)
    self.txtFP.grid(row = 9, column = 1)
    self.txtFP.insert(0, "100")

  def addOutput(self):
    """ add button and output elements """
    self.btnCalc = Button(self, text = "calculate grade")
    self.btnCalc.grid(row = 10, columnspan = 2)
    self.btnCalc["command"] = self.calculate

    Label(self, text = "Lab Percent").grid(row = 11, column = 0)
    # more attributes can be set in constructor:
    # bg = background color, fg = foreground color
    # anchor determines alignment ("nsew")
    # relief is border
    self.lblLabs = Label(self, bg = "#fff", anchor = "w", relief = "groove")
    #sticky propery in grid changes width and height
    self.lblLabs.grid(row = 11, column = 1, sticky = "we")

    Label(self, text = "Exam Percent").grid(row = 12, column = 0)
    self.lblExams = Label(self, bg = "#fff", anchor = "w", relief = "groove")
    self.lblExams.grid(row = 12, column = 1, sticky = "we")

    Label(self, text = "Overall Percent").grid(row = 13, column = 0)
    self.lblTotal = Label(self, bg = "#fff", anchor = "w", relief = "groove")
    self.lblTotal.grid(row = 13, column = 1, sticky = "we")

  def calculate(self):
    """ calculate the grades """

    #get lab average
    lab1 = int(self.txtLab1.get())
    lab2 = int(self.txtLab2.get())
    lab3 = int(self.txtLab3.get())
    lab4 = int(self.txtLab4.get())
```

```
    labTot = lab1 + lab2 + lab3 + lab4
    labPerc = labTot / 4.0

    self.lblLabs["text"] = "%.2f" % labPerc

    #get exam average
    mt = int(self.txtMT.get())
    fe = int(self.txtFE.get())

    examPerc = (mt + fe) / 2.0
    self.lblExams["text"] = "%.2f" % examPerc

    #project percentage needs no more calculation
    projPerc = int(self.txtFP.get())

    #calculate total percentage
    total = (labPerc * .4) + (examPerc * .4) + (projPerc * .2)
    self.lblTotal["text"] = "%.2f" % total

def main():
  app = App()
  app.mainloop()

if __name__ == "__main__":
  main()
```