In this problem we are supposed to create a python desktop calendar app with the following features:

1. Calendar of one month should be displayed on app window
2. Present date has to be highlighted
3. Help option has to be available to know the working of the app
4. The app should be able to display the calendar of any month (-12 to +12) count away from the present month

The app is made in python3. And the command to run the app is

```
$python3 <filename.py> <optional>
```

<optional> = <Integer>

can be any integer which represents the count from the present month of the month to be displayed (valid range according to question -12 to +12), for any value not in range the app outputs the line requesting to give the integer in the valid range.

= ‘-help’

When the initial run has ‘-help’ as runtime argument, the app displays it working.

App options:

App has 2 buttons(Help, Calendar) and one entry box. Entry box is for entering the integer by which the present month has to be forwarded or counted back to the month required to be displayed. And on pressing the “Calendar” button after entering the integer, the app displays the desired month with the present date highlighted. “Help” button when pressed at any time displays the working of the app to the user. The valid range for the integer being entered is -12 to +12, for entries out of range the app requests the user to give the input in the valid range.

CODE PYTHON3

```
from tkinter import *
import sys
import re
import calendar
from datetime import datetime
from math import ceil

# This function is for getting the calendar of the month to be displayed
from the calendar module of python
```

```
# We store the output in a string which will be later processed to find
the current day's position in order to
# add the tag, so that the present day can be highlighted while being
displayed on the app
# Function returns the string to be displayed on app
def printy(current_year, current_month):
   i =0
   s = calendar.month(current_year, current_month)
   t=''
   ans = ''
   # checking through the calendar output string to make our string
   for i in s:
       if i!=' ':
           t+=i
       else:
           if t=="":
               ans+=i
           else:
               try:
                   if int(t)==today.day:
                       ans+= t
                       t=''
                   else:
                       ans+=t
               except:
                   ans+=t
               t=""
               ans+=i

   if t=="":
       ans+=i
   else:
       try:
           if int(t)==today.day:
               ans+= t
               t=''
```

```
            else:
                ans+=t
        except:
            ans+=t
        t=""
        ans+=i

    return ans

class MyWindow:

    def __init__(self, win, initial_var):
        self.lbl1=Label(win, text='Enter a number')
        self.t1=Entry(bd=3)
        self.lbl1.place(x=30, y=50)
        self.t1.place(x=150, y=50)
        self.b1= Button(win, text='Calendar', command=self.cal)
        self.b2= Button(win, text='Help' , command=self.help)
        self.b1.place(x=100, y=100)
        self.b2.place(x=230, y=100)

        # Initialising the scrollbar on app window

        self.scroll_bar = Scrollbar(win)
        self.scroll_bar.place(x=350,y=150,height= 200)

        # Using text widget for displaying the output
        self.t3 = Text(win, yscrollcommand = self.scroll_bar.set )
        self.t3.place(x=50,y=150,height=200,width=300)
        self.scroll_bar.config( command = self.t3.yview)
        self.s_tag= ["1.0","1.0"]

        # These check if there are any initial runtime arguments in order
to first deal with them accordingly
        if(initial_var == 0):
            self.cal(0)
            initial_var= "Done"
        elif(initial_var == 'h'):
            self.help()
            initial_var = "Done"
        elif(isinstance(initial_var,int)):
            self.cal(initial_var)
            initial_var = "Done"

    # This functions is triggered when the "Help" button is pressed by
the user, or when '-help' option is used in the initial runtime
arguments
    # This function displays the working of the app on the app window
    def help(self):
        self.t3.delete(1.0, END)
        info = "\n  Welcome to Python Calendar App\n"+ "\nThis app shows
in the calendar, the present month and year with the day highlighted. "
        info+= "On entering a number, the app shows you the month
obtained on moving from the present month by the given count.\n"
        info+= "\nSteps:\n1.Enter a number between -12 to 12\n2.Press
'Calendar'\n"
        self.t3.insert(END, info)

    # This function is triggered when the "Calendar" button is pressed by
the user, or when there is an integer in runtime arguments.
    # This function displays the month required with present day
highlighted on the app window
    def cal(self,num1=None):
        self.t3.delete(1.0, END)

        # checks if the value entered is in valid range
        if(num1 == None):
            try:
                temp = self.t1.get()
                try:
                    int(temp)
                    num1= int(temp)
```

```
               except ValueError:
                   num1 = -16
           except:
               num1=0

       # if the input is not in range, app displays that the value
should be in valid range
       if(num1<-12 or num1>12):
           info="\nNumber entered should be in between -12 and 12\n"
           self.t3.insert(INSERT, info)

       else:
           # Getting the values of present day, month, year from python
modules
           today = datetime.today()
           current_month = today.month
           current_year = today.year
           current_date = str(today.day)

           # going to the required month from present month given from
input integer
           current_month = current_month+ num1
           if current_month<=0:
               current_month = 12 + current_month
               current_year = current_year-1
           if current_month>12:
               current_month = current_month-12
               current_year = current_year+1

           calendar.setfirstweekday(calendar.SUNDAY)
           #  getting the string to be displayed on app as the month
           ans= printy(current_year, current_month)
           afteryr = re.search(str(current_year),ans).end()
           cr = afteryr
           # getting the positions of present date to set the tag
           st = re.search(current_date,ans[cr:]).start()
```

```
            en = re.search(current_date,ans[cr:]).end()
            while st < afteryr:
                cr=en
                st =re.search(current_date,ans[cr:]).start()
                en= re.search(current_date,ans[cr:]).end()

            st += cr
            en += cr

            self.s_tag[0] ="1.0+" + str(st) + "c"
            self.s_tag[1] ="1.0+" + str(en) + "c"

            # printing the month
            self.t3.insert(END, ans)

            # highlighting the present date
            self.t3.tag_add("hl", self.s_tag[0], self.s_tag[1])
            self.t3.tag_configure("hl",background="black",
foreground="white")

args_list= sys.argv

# checks if there are any initial runtime arguments
if(len(args_list)==1):
   initial_var = 0
else:
   if(args_list[1]== "-help"):
       initial_var = 'h'
   else:
       initial_var = int(args_list[1])

window=Tk()
mywin=MyWindow(window,initial_var)
window.title('Python Calender App')
window.geometry("400x400+10+10")
window.mainloop()
```