```
#Minecraft cyclic backup program by Ben Cooper.
#Written according to pep8 standards
#Designed for Python 3.3.2
#
#os is used to change the directory of the script
#shutil is used for the copytree command to copy folders
#time is to append timestamps to backup sub folders
#tkinter is used to create the gui
import os
import shutil
import time
from tkinter import *


def backup_start(filepath, savepath, seconds):
        '''(str, str, int)
        Check for errors in users input using exception handling then
        returns custom error message via label.
        Call backup_cycle iff no exceptions raised
        '''
        try:
                #The paths are checked for validity here and seconds
                # is being checked for being a positive number
                os.chdir(filepath)
                os.chdir(savepath)
                seconds = int(seconds)*1000
                if seconds <= 0:
                        raise ValueError
        except FileNotFoundError:
                #if the paths are invalid then label to changed to inform
                # the user
                num.set('Invalid path!')
                root.update_idletasks()
        except ValueError:
                #if seconds is not a real number then the label is changed to
                # inform the user
                num.set('Invalid time!')
                root.update_idletasks()
        else:
                #if everything is ok then the function makes a new directory
                # and adds it to the save path.  It then calls bakcup_cycle
                # after 'seconds' second(s)
                os.system('md Minebackup')
                savepath += '\\Minebackup'
                backup_cycle(filepath, savepath, seconds)


def backup_cycle(filepath, savepath, seconds):
        '''(str, str, str)
        Creates the next backup iff there are no errors otherwise exit.
        Call backup_cycle again once completed
        '''
        #counter was made as a global variable to prevent it from
```

```
        # re-initializing when the function is called again.
        #global is used so the function can edit the global variable
        global counter
        try:
                #Increments counter and then updates label to reflect current
                # backup number.
                counter += 1
                num.set('Current Backup: ' + str(counter))
                root.update_idletasks()
                #curpath represents the custom name of the sub folders.
                # It includes the backup number and a time stamp
                curpath = '\\Backup' + str(counter) + time.strftime(
                    " %I.%M")
                #The file (filepath) is then copied to the new directory with
                # the name curpath
                shutil.copytree(filepath, savepath+curpath)
                #After 'seconds' second(s) this function calls itself again
                root.after(seconds, lambda: backup_cycle(
                    filepath, savepath, seconds))
        except:
                #The program quits if there is an error
                exit()
if __name__ == '__main__':
        #Initializing the counter variable
        counter = 0
        #This block of code creates the user interface in tkinter.
        #The window is 600x100 and is not resizeable in any dimension
        root = Tk()
        root.geometry("600x100")
        root.resizable(0, 0)
        #changing the size of the second column (column 1)
        root.columnconfigure(1, weight=1)
        #settings a special text variable for the output label (currnum)
        num = StringVar()
        num.set('Current Backup: 0')
        #adding a custom title to the window
        root.title('Minebackup')
        #creating the information labels and text boxes
        filel = Label(root, text='World Path: ')
        filet = Entry(root)
        savel = Label(root, text='Backup Path: ')
        savet = Entry(root)
        timel = Label(root, text='Interval (seconds): ')
        timet = Entry(root)
        #this output label uses num as its text variable
        # this allows different functions to edit the label
        currnum = Label(root, textvariable=num)
        #this button starts the calculations by calling backup_start
        startbut = Button(root, text='Start Cycle!', command=lambda:
                          backup_start(filet.get(), savet.get(), timet.get()))
        #this button allows the user to quit early by stopping the main window
        endbut = Button(root, text="Quit", command=root.destroy)
        #this block tells the widgets where to go
```

```
        filel.grid(row=0)
        filet.grid(row=0, column=1, sticky=E+W)
        savel.grid(row=1)
        savet.grid(row=1, column=1, sticky=E+W)
        timel.grid(row=2)
        timet.grid(row=2, column=1, sticky=W)
        startbut.grid(row=3)
        endbut.grid(row=3, column=1, sticky=W)
        currnum.grid(row=3, column=1)
        #starts the graphical user interface
        root.mainloop()
```