

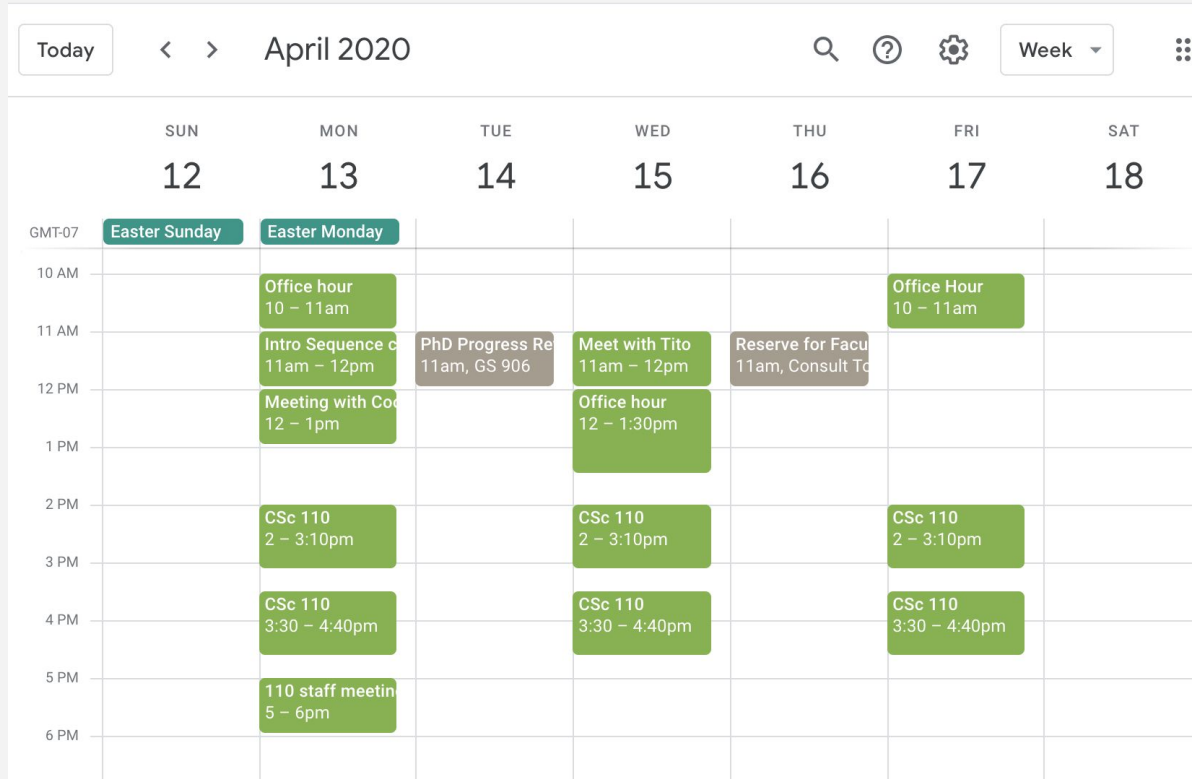
# CSc 110

## Combining Data Structures

Benjamin Dicken



# Google Calendar




# Weekly Planner Book



# Weekly Planner Program

Running: week\_planner.py

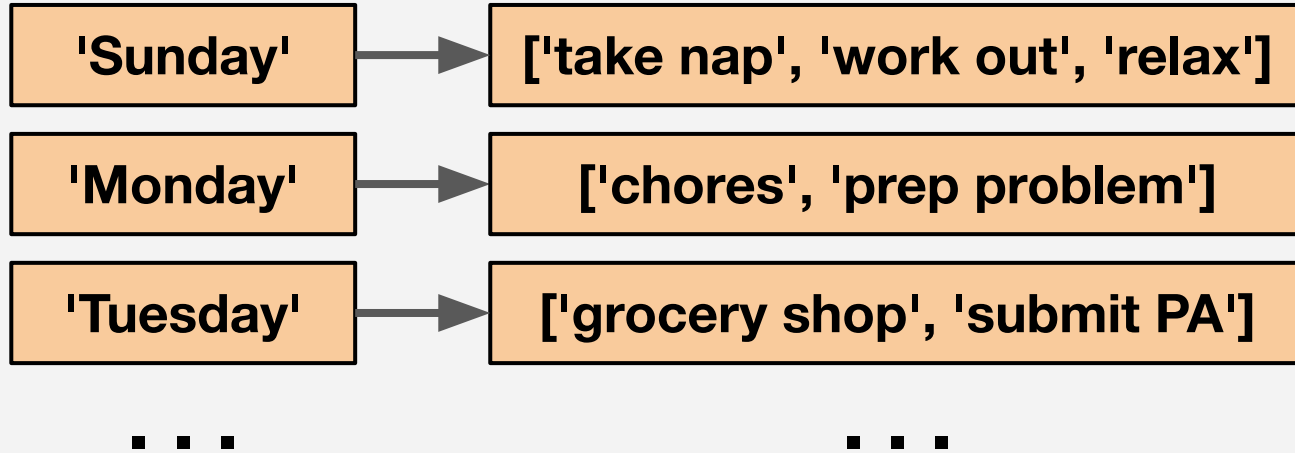
```
Day of week for task: Sunday
Task for Sunday: take nap
Day of week for task: Friday
Task for Friday: get take-out
Day of week for task:
```

Python 

weekly planner

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
take nap	prep problem	turn in PA	prep problem grocery shop		prep problem get take-out	

# Dictionary Mapping String to List



```
import graphics
```

```
DAYS_IN_ORDER = ['Sunday', 'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday']
```

```
def draw_calendar_view(gui, tasks):
```

```
    ''' Draw a view that shows the days of the week and the tasks  
    ...
```

```
def get_input():
```

```
    ''' Get a day and task input from the user, return them both.  
    ...
```

```
def main():
```

```
    ''' (A) Create an dictionary to store the tasks.  
        Should map day string to a list of tasks  
        (B) Create canvas  
        (C) Repeatedly get user input, and update canvas after each one  
    ...
```

```
main()
```

# Implement get\_input

```
def get_input():  
    ''' Get a day and task input from the user, return them both.  
        (A) Get the day input from the user  
        (B) If the input is 'exit' then just return 'exit'  
        (C) Get the task string  
        (D) return both of the strings (use multiple return)  
    ...  
'''
```

# The get\_input function

```
def get_input():  
    day = input('Day of week for task: ')  
    if day == 'exit':  
        return 'exit', 'exit'  
    task = input('Task for ' + day + ': ')  
    return day, task
```



# Implement draw\_calendar\_view

```
def draw_calendar_view(gui, tasks):
    gui.clear()
    location = 25
    width = 150
    height = 200
    border = 25
    for day in DAYS_IN_ORDER:
        gui.rectangle(location, border, width, height, 'blue')           # Border
        gui.rectangle(location+2, border+2, width-4, height-4, 'white') # Fill
        gui.text(location+10, border + 10, day, 'dark green', border)   # Label
        gui.line(location+2, 70, location+width-2, 70, 'red', 3)       # Separator
        # TODO: draw the tasks!
        location += width
    gui.update_frame(60)
```

```
def draw_calendar_view(gui, tasks):
    gui.clear()
    location = 25
    width = 150
    height = 200
    border = 25
    for day in DAYS_IN_ORDER:
        gui.rectangle(location, border, width, height, 'blue')           # Border
        gui.rectangle(location+2, border+2, width-4, height-4, 'white') # Fill
        gui.text(location+10, border + 10, day, 'dark green', border)   # Label
        gui.line(location+2, 70, location+width-2, 70, 'red', 3)       # Separator
        task_offset = 80
        for task in tasks[day]:                                           # Loop thru tasks
            gui.text(location+10, task_offset, task, 'black', 15)       # Task
            task_offset += 20
        location += width
    gui.update_frame(60)
```

# Implement main

```
def main():  
    ''' (A) Create the graphics canvas  
        (B) Create the dictionary that will map strings (days of  
            the week) to a list (the list of tasks for that day)  
        (C) Create a loop to repeatedly get input from the user  
        (D) On each iteration, get input, then update the canvas  
    ...
```

# The main function

```
def main():
    gui = graphics.graphics(1100, 250, 'weekly planner')
    tasks = {}
    for day in DAYS_IN_ORDER:
        tasks[day] = []
    draw_calendar_view(gui, tasks)
    while True:
        day, task = get_input()
        if day == 'exit':
            return
        tasks[day].append(task)
        draw_calendar_view(gui, tasks)
```

```

import graphics

DAYS_IN_ORDER = ['Sunday', 'Monday', 'Tuesday',
                 'Wednesday', 'Thursday', 'Friday', 'Saturday']

def draw_calendar_view(gui, tasks):
    """
    This function's job is to draw the days of the week in a calendar
    view, and putting the tasks on each day.
    gui: the graphics object
    tasks: dictionary of tasks
    This function does not need to return anything.
    """
    gui.clear()
    location = 25
    width = 150
    height = 200
    border = 25
    for day in DAYS_IN_ORDER:
        gui.rectangle(location, border, width, height, 'blue')
        gui.rectangle(location+2, border+2, width-4, height-4, 'white')
        gui.text(location+10, border + 10, day, 'dark green', border)
        gui.line(location+2, 70, location+width-2, 70, 'red', 3)
        task_offset = 80
        for task in tasks[day]:
            gui.text(location+10, task_offset, task, 'black', 15)
            task_offset += 20
        location += width
    gui.update_frame(60)

```

```

def get_input():
    """
    This function does:
    (A) Get the day input from the user
    (B) If the input is 'exit' then just return 'exit'
    (C) Get the task string
    (D) return both of the strings (use multiple return)
    Returns two strings, a day and task
    """
    day = input('Day of week for task: ')
    if day == 'exit':
        return 'exit', 'exit'
    task = input('Task for ' + day + ': ')
    return day, task

def main():
    gui = graphics.graphics(1100, 250, 'weekly planner')
    tasks = {}
    for day in DAYS_IN_ORDER:
        tasks[day] = []

    draw_calendar_view(gui, tasks)

    while True:
        day, task = get_input()
        if day == 'exit':
            return
        tasks[day].append(task)
        draw_calendar_view(gui, tasks)

main()

```

# Other Ideas

- Saving the tasks to a file
- Loading the tasks from a file
- Wrapping task lines
- Inputting the tasks via the canvas, rather than console