

# OOF (Outstanding Organisation Friend) - User Guide

1. Introduction .....	2
1.1. What is <b>OOF</b> ? .....	2
1.2. What can <b>OOF</b> do? .....	3
1.3. How does <b>OOF</b> address our target audience? .....	3
1.4. What is this guide for? .....	3
2. Quick Start .....	3
3. Features .....	4
3.1. Viewing our manual: <b>help</b> .....	5
3.2. Viewing the usage of individual commands: <b>help</b> .....	6
3.3. Adding a task: <b>deadline</b> .....	6
3.4. Adding an event: <b>event</b> .....	7
3.5. Adding a task: <b>todo</b> .....	7
3.6. Setting a recurring task: <b>recurring</b> .....	8
3.7. Listing tasks: <b>list</b> .....	10
3.8. Marking task as done: <b>done</b> .....	11
3.9. Deleting a task: <b>delete</b> .....	12
3.10. Finding tasks quickly: <b>find</b> .....	12
3.11. Choosing a threshold for tasks: <b>threshold</b> .....	13
3.12. Viewing a summary of a day's task by date: <b>schedule</b> .....	13
3.13. Viewing a summary of the next day's task: <b>summary</b> .....	14
3.14. Viewing free time slots: <b>free</b> .....	14
3.15. Viewing tasks in week view: <b>viewweek</b> .....	17
3.16. Viewing all tasks in calendar view: <b>calendar</b> .....	18
3.17. Setting reminders for upcoming deadlines: <b>NIL</b> .....	18
3.18. Semesters .....	19
3.19. Modules .....	21
3.20. Lessons .....	23
3.21. Adding assessment data: <b>assessment</b> .....	24
3.22. Adding assignment data: <b>assignment</b> .....	25
3.23. Start Assignment Tracker: <b>start</b> .....	25
3.24. Pause Assignment Tracker: <b>pause</b> .....	26
3.25. Stop Assignment Tracker: <b>stop</b> .....	26
3.26. View Assignment Tracker: <b>viewTracker</b> .....	26
3.27. Exiting the program: <b>bye</b> .....	27
3.28. View undone tasks brought forward to the next day: <b>undone</b> [coming soon in v2.0] .....	27
3.29. Filter tasks by categories: <b>filter</b> [coming soon in v2.0] .....	27

3.30. Adding a task: <b>tentative</b> [coming soon in v2.0] .....	28
3.31. Adding a task: <b>do-after</b> [coming soon in v2.0] .....	28
3.32. Add estimated time taken: <b>estimate</b> [coming soon in v2.0] .....	28
3.33. Adding a task: <b>range</b> [coming soon in v2.0] .....	29
3.34. View two different calendars side-by-side: <b>viewDual</b> [coming soon in v2.0] .....	29
3.35. Export calendar: <b>export</b> [coming soon in v2.0] .....	29
4. FAQ .....	29
5. Command Summary .....	29
5.1. Available Commands .....	30
5.2. Coming Soon .....	32

By: **Team W17-4** Since: **Aug 2019** Licence: **MIT**

# 1. Introduction

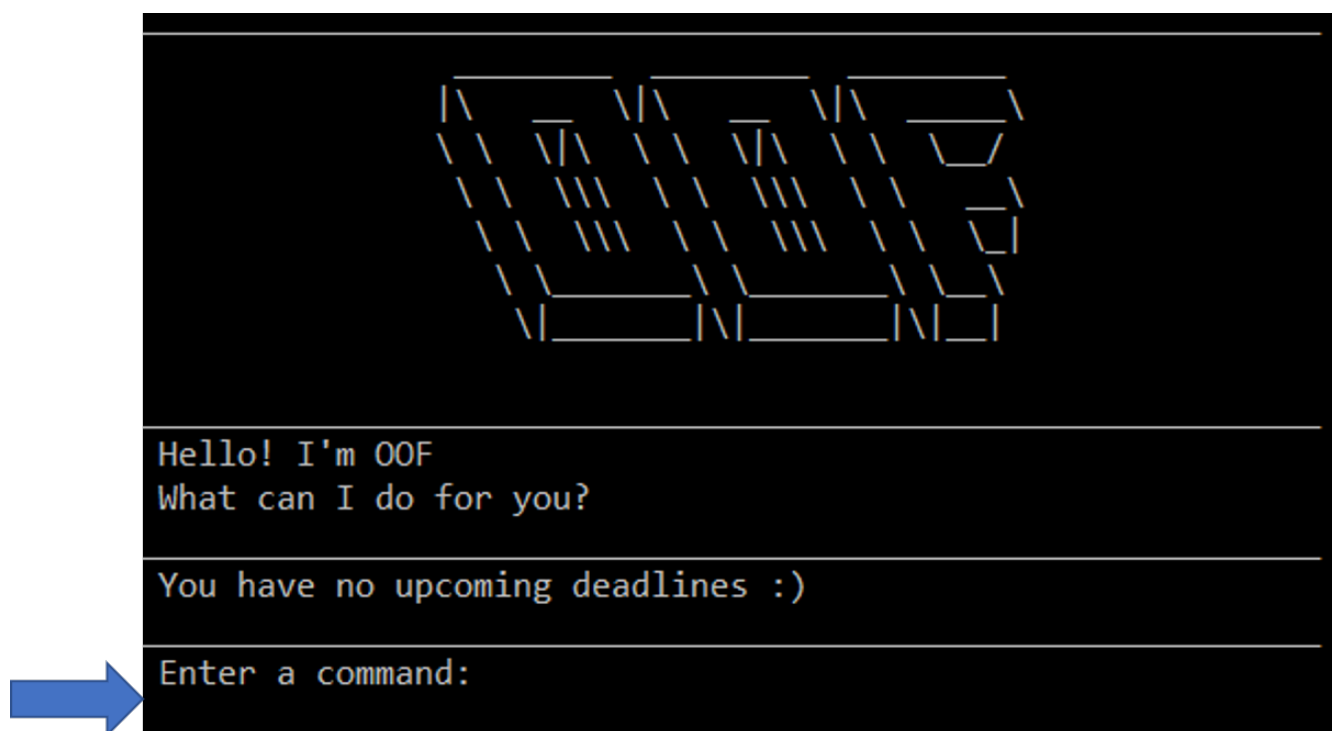


Figure 1. OOF welcome screen

## 1.1. What is OOF?

**OOF** (Oustanding Organisational Friend) is a Command Line Interface (CLI) program that allows you to save your tasks, assignments, modules taken, etc. **OOF** is catered towards university students who want to use a desktop application to manage their tasks in a friendly and efficient manner. **OOF** is optimized for users who prefer to work with the CLI while still reaping the benefits of a Graphical User Interface (GUI).

## 1.2. What can OOF do?

Besides saving your tasks very effectively in persistent storage, **OOF** allows your tasks to be displayed in friendly formats such as calendar format or a tabular format where your tasks are sorted chronologically for any particular week. You can also let **OOF** remind you of tasks that are expiring based on a customisable threshold. On top of that, you can track your progress and see if you are on track by using our tracking feature.

## 1.3. How does OOF address our target audience?

Most university students are often busy and **OOF** aims to reduce the time students spend on managing their tasks. **OOF** allows students to enter one-liner commands quickly into our program and hence spend less time logging down the tasks to be done. Furthermore, **OOF** allows tasks to be viewed in insightful formats and also provides timely reminders for tasks with their deadline nearing.

## 1.4. What is this guide for?

This guide aims to educate you on how to use our application by providing example usages of all its features. The features can be found in [Section 3, “Features”](#) section.

Interested in using **OOF** to plan your timetable more effectively? Jump to [Section 2, “Quick Start”](#) to get started! Enjoy!

# 2. Quick Start

1. Ensure you have Java 11 or above installed on your computer.
2. Download the latest release. [here](#).
3. Copy the files to the folder you want to use as the home folder for your Outstanding Organization Friend.
4. Run the command “java -jar oof.jar”. The application should load within a few seconds.

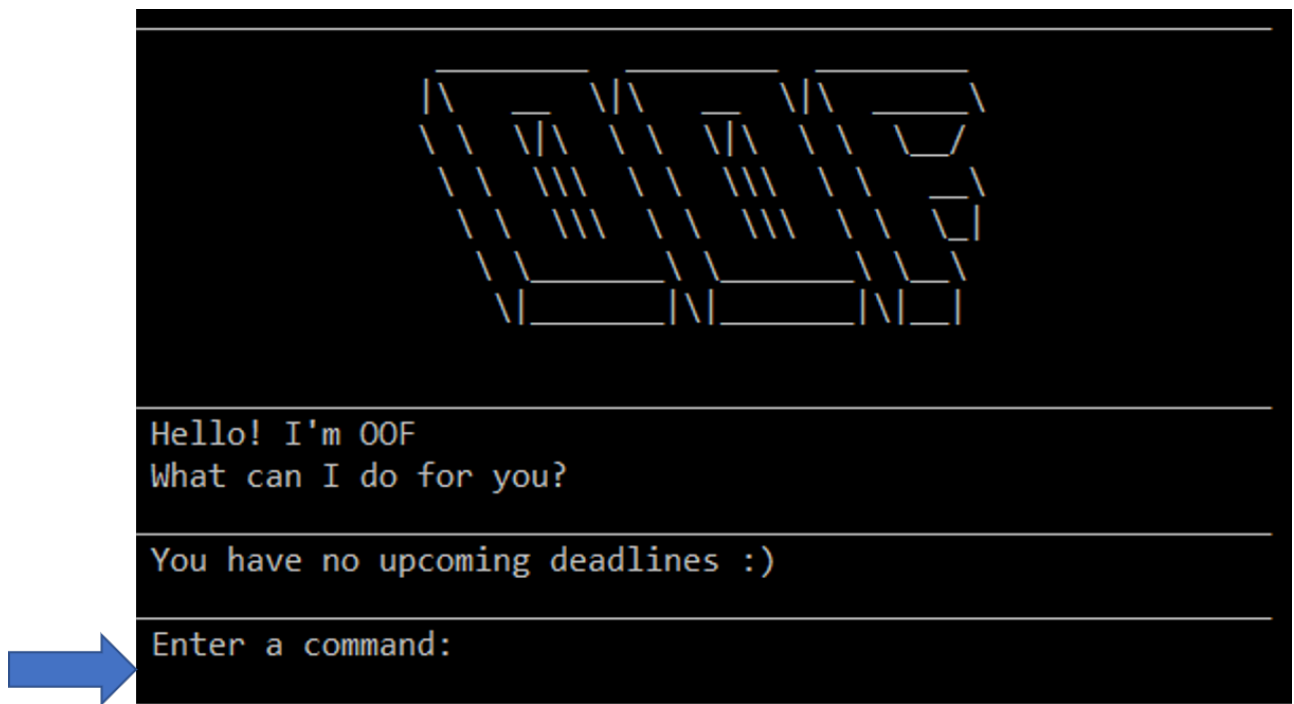


Figure 2. OOF welcome screen

5. Type a task description in the terminal and press `Enter` to run it.  
e.g. typing `help` and pressing `Enter` will list the commands present.
6. Some example commands you can try:
  - `deadline homework /by 12-12-2019 11:11` : adds a task called `homework` to the saved tasks with the deadline `12-12-2019 11:11`.
  - `calendar` : displays all saved tasks in a calendar view.
  - `Bye` : exits the application.

A summary of all the features available in **OOF** can be found in [Section 5, “Command Summary”](#).

Refer to [Section 3, “Features”](#) for details of each command.

## 3. Features

In this section, the expected command format will be introduced, and you can expect to learn the various commands you can use.

### Command Format

- Words in `UPPER_CASE` are the parameters to be supplied by the user e.g. `deadline DESCRIPTION /by DD-MM-YYYY HH:MM`



Don't worry if you do not understand everything at once.

There are plentiful examples provided to aid your understanding of the commands' usage.

## 3.1. Viewing our manual: help

Shows you a list of commands that can be used.

Format: help

Example:

- User enters help

```
Enter a command:
help
===== OOF MANUAL =====

NAME
    OOF -- Outstanding Organisation Friend

DESCRIPTION
    The following options are available:

Deadline      deadline DESCRIPTION /by DD-MM-YYYY HH:MM
Event          event DESCRIPTION /from DD-MM-YYYY HH:MM /to DD-MM-YYYY HH:MM
Todo           todo DESCRIPTION /on DD-MM-YYYY
Do-after       do-after INDEX DESCRIPTION
Recurring      recurring DESCRIPTION
Tentative      tentative DESCRIPTION
List           list
Done           done INDEX
Delete         delete INDEX
Find           find DESCRIPTION
Filter         filter CATEGORY
Threshold      threshold HH
Color Code     colorcode INDEX #RRGGBB
Schedule       schedule DD-MM-YYYY
Summary        summary
Sort           sort
View Undone    viewUndone
Free           free DD-MM-YYYY
View Week      viewWeek
```

Figure 3. Output of Help Command

Usage of all the features is shown to you if help is entered.

## 3.2. Viewing the usage of individual commands: **help**

Shows you the specific usage for the command you have entered.

Format: **help** COMMAND

Example:

- **help** Deadline

```
Enter a command:
help deadline

Deadline          deadline DESCRIPTION /by DD-MM-YYYY HH:MM

Enter a command:
```

Figure 4. Example of **help** COMMAND usage

Correct syntax of adding a **deadline** is shown.

## 3.3. Adding a task: **deadline**

You can choose to add a task with a deadline.

Format: **deadline** DESCRIPTION /by DD-MM-YYYY HH:MM

- **Description** of the task to be done can have multiple words, not just limited to single-word descriptions.
- **Date and time** have to **strictly** be in the format as stated above.

Example:

- **deadline** homework /by 2019 20-11-2019 13:00

```
Enter a command:
deadline homework /by 2019 20-11-2019 13:00

Got it. I've added this task:
      [D][N] homework (by: 20-11-2019 13:00)
Now you have 22 tasks in your list.

Enter a command:
```

Figure 5. Example usage of **deadline** feature

Adds a task with description and datetime to be **homework** and **2019 20-11-2019 13:00** respectively.

### 3.4. Adding an event: **event**

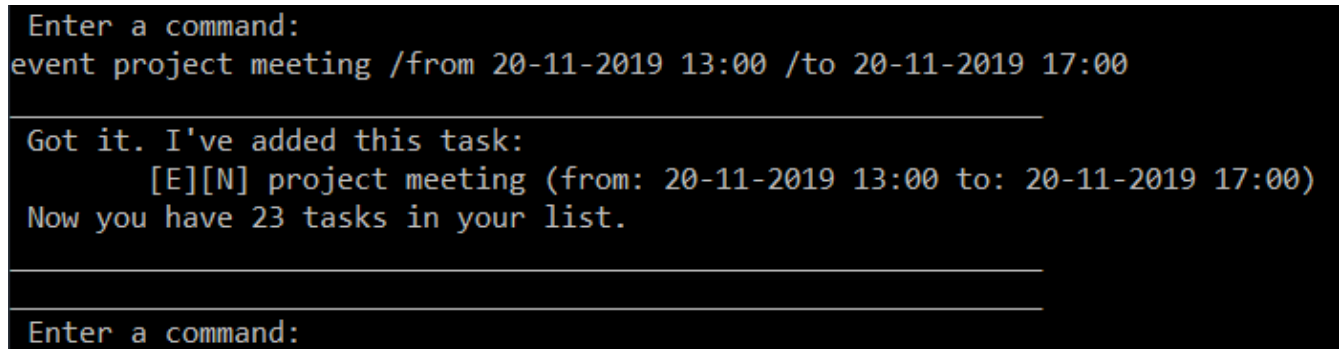
You can add an event with a scheduled starting and ending time.

Format: **event** DESCRIPTION /from DD-MM-YYYY HH:MM /to DD-MM-YYYY HH:MM

- **Description** of the task to be done can have multiple words, not just limited to single-word descriptions.
- **Date and time** have to **strictly** be in the format as stated above.

Example:

- **event project meeting /from 20-11-2019 13:00 /to 20-11-2019 17:00**

A terminal window with a black background and light green text. The prompt 'Enter a command:' is shown. The user enters 'event project meeting /from 20-11-2019 13:00 /to 20-11-2019 17:00'. The terminal responds with 'Got it. I've added this task:' followed by '[E][N] project meeting (from: 20-11-2019 13:00 to: 20-11-2019 17:00)' and 'Now you have 23 tasks in your list.' The prompt 'Enter a command:' is shown again.

```
Enter a command:
event project meeting /from 20-11-2019 13:00 /to 20-11-2019 17:00

Got it. I've added this task:
      [E][N] project meeting (from: 20-11-2019 13:00 to: 20-11-2019 17:00)
Now you have 23 tasks in your list.

Enter a command:
```

Figure 6. Example usage of event feature

Adds an event with description, start and end time to be **project meeting, 20-11-2019 13:00** and **20-11-2019 17:00** respectively.

### 3.5. Adding a task: **todo**

You can choose to add a task to be done on a specific day.

Format: **todo** DESCRIPTION /on DD-MM-YYYY

- **Description** of the task to be done can have multiple words, not just limited to single-word descriptions.
- **Date** has to **strictly** be in the format as stated above.

Example:

- **todo withdraw money /on 19-11-2019**

```
Enter a command:
todo withdraw money /on 19-11-2019

Got it. I've added this task:
      [T][N] withdraw money (on: 19-11-2019)
Now you have 24 tasks in your list.

Enter a command:
```

Figure 7. Example usage of todo feature

Adds a task called `withdraw money` on `19-11-2019`.

### 3.6. Setting a recurring task: `recurring`

You can select a task that will be repeated based on your preference.

Format: `recurring INDEX NUMBER_OF_OCCURRENCES`

- The `INDEX` refers to the index number displayed in the list of tasks recorded. (`list` can be used to display the saved tasks).
- `NUMBER_OF_OCCURRENCES` refers to the number of times the selected task recurs. \*The user chooses to enter a `FREQUENCY` which is an option from 1-4.



The task will require you to enter the frequency of recurrence in this manner afterward:

1. DAILY
2. WEEKLY
3. MONTHLY
4. YEARLY

Example:

1. The user enters `recurring 4 3`



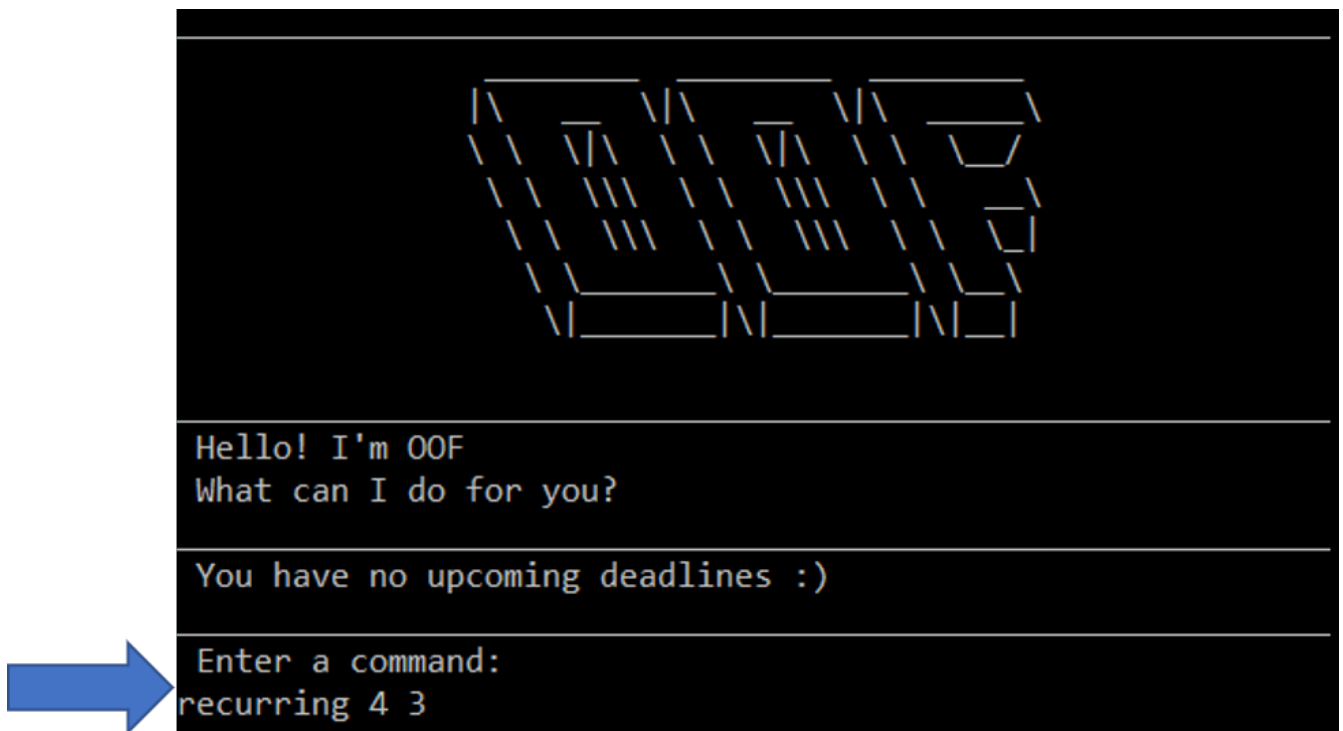


Figure 8. Example to show recurring feature's usage

2. The user presses `ENTER` and he/she can then choose an option by entering a number `1-4` to choose the frequency of recurrence.

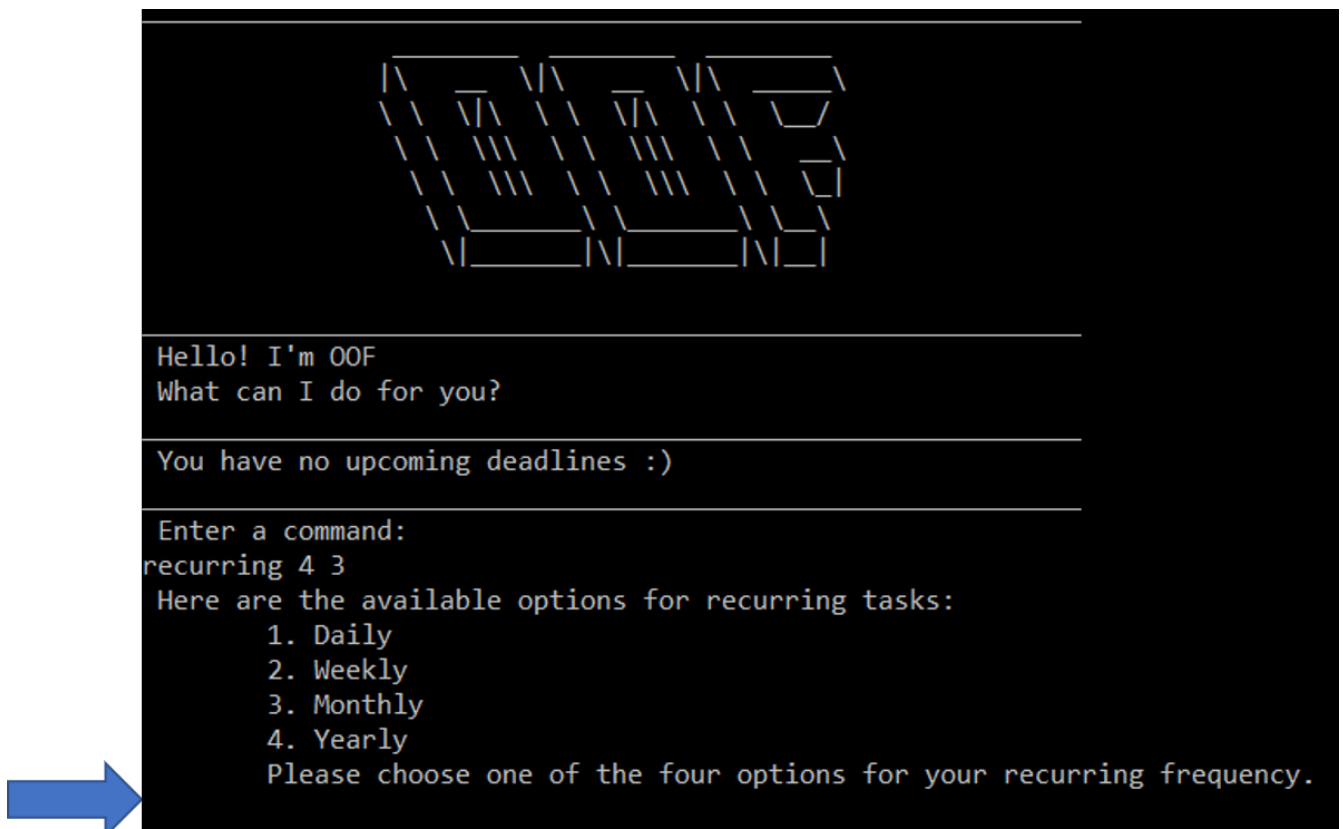


Figure 9. Options for recurring frequency

+3.+The user chooses option `2`.

```

recurring 4 3
Here are the available options for recurring tasks:
  1. Daily
  2. Weekly
  3. Monthly
  4. Yearly
Please choose one of the four options for your recurring frequency.

2

```

---

```

I have added recurring tasks:

```

---

```

Here are the tasks in your list:
  1. [T][Y] borrow another book (on: 13-10-2019)
  2. [D][N] homework (by: 13-10-2019 23:59)
  3. [E][N] lecture (from: 08-10-2019 10:00 to: 08-10-2019 12:00)
  4. [E][N] tutorial (from: 09-10-2019 17:00 to: 09-10-2019 18:00)
  5. [E][N] test (from: 10-10-2019 09:00 to: 10-10-2019 10:00)
  6. [T][N] cs2105 cs2106 cs2107 cs2113t cs2101 (on: 13-10-2019)
  7. [D][N] homework (by: 14-10-2019 23:59)
  8. [E][N] steamboat (from: 15-10-2019 18:00 to: 15-10-2019 20:00)
  9. [D][N] homework (by: 29-10-2019 23:59)
 10. [D][N] homework (by: 14-10-2019 10:00)
 11. [E][N] tutorial (from: 16-10-2019 17:00 to: 16-10-2019 18:00)
 12. [E][N] tutorial (from: 23-10-2019 17:00 to: 23-10-2019 18:00)
 13. [E][N] tutorial (from: 30-10-2019 17:00 to: 30-10-2019 18:00)

```

---

```

Enter a command:

```

Figure 10. Output after selecting option 2

## 3.7. Listing tasks: `list`

You can list all the tasks that you have saved in **OOF**.

Format: `list`

Example:

- User enters `list`

```
Enter a command:
list

Here are the tasks in your list:
 1. [T][Y] borrow another book (on: 13-10-2019)
 2. [D][N] homework (by: 13-10-2019 23:59)
 3. [E][N] lecture (from: 08-10-2019 10:00 to: 08-10-2019 12:00)
 4. [E][N] tutorial (from: 09-10-2019 17:00 to: 09-10-2019 18:00)
 5. [E][N] test (from: 10-10-2019 09:00 to: 10-10-2019 10:00)
 6. [T][N] cs2105 cs2106 cs2107 cs2113t cs2101 (on: 13-10-2019)
 7. [D][N] homework (by: 14-10-2019 23:59)
 8. [E][N] steamboat (from: 15-10-2019 18:00 to: 15-10-2019 20:00)
 9. [D][N] homework (by: 29-10-2019 23:59)
10. [D][N] homework (by: 14-10-2019 10:00)
11. [E][N] tutorial (from: 16-10-2019 17:00 to: 16-10-2019 18:00)
12. [E][N] tutorial (from: 23-10-2019 17:00 to: 23-10-2019 18:00)
13. [E][N] tutorial (from: 30-10-2019 17:00 to: 30-10-2019 18:00)
14. [D][N] complete lab assignment (by: 30-10-2019 23:59)
15. [T][N] go to make up lecture (on: 29-10-2019)
16. [T][N] go to lecture (on: 27-10-2019)
17. [D][N] complete tutorial (by: 27-10-2019 23:59)
18. [D][N] complete DG (by: 27-10-2019 23:59)
19. [D][N] complete UG (by: 28-10-2019 23:59)
20. [D][N] tutorial (by: 30-10-2019 14:00)
21. [D][N] lecture (by: 30-10-2019 16:00)
22. [D][N] homework (by: 20-11-2019 13:00)
23. [E][N] project meeting (from: 20-11-2019 13:00 to: 20-11-2019 17:00)
24. [T][N] withdraw money (on: 19-11-2019)

Enter a command:
```

Figure 11. Output of list command

A list of tasks currently saved in OOF will be displayed.

### 3.8. Marking task as done: done

You can mark tasks as completed so that you can track your progress.

Format: done INDEX

- The INDEX refers to the index number displayed in the list of tasks recorded. (list can be used to display the saved tasks).

Examples:

- done 2

```
Enter a command:
done 2

Nice! I've marked this task as done:
      [D][Y] homework (by: 13-10-2019 23:59)

Enter a command:
```

Figure 12. Output of done command.

Deletes the 1st task in the list of tasks.

### 3.9. Deleting a task: delete

You can delete tasks that you have completed or no longer valid.

Format: **delete** INDEX

- The **INDEX** refers to the index number displayed in the list of tasks recorded. (**list** can be used to display the saved tasks).

Examples:

- **delete 10**

```
Enter a command:
delete 10

Noted. I've removed this task:
      [D][N] homework (by: 14-10-2019 10:00)
Now you have 23 tasks in your list.

Enter a command:
```

Figure 13. Output of delete command

Deletes the 10th task in the list of tasks.

### 3.10. Finding tasks quickly: find

You can quickly find anything you have inputted by providing **OOF** with a keyword.

Format: **find** DESCRIPTION

- **Description** of the task to be done can have multiple words, not just limited to single-word descriptions.

Example:

- **find complete**

```
Enter a command:
find complete

Here are the matching tasks in your list:
  1. [D][N] complete lab assignment (by: 30-10-2019 23:59)
  2. [D][N] complete tutorial (by: 27-10-2019 23:59)
  3. [D][N] complete DG (by: 27-10-2019 23:59)
  4. [D][N] complete UG (by: 28-10-2019 23:59)

Enter a command:
```

Figure 14. Output of *find* command

Finds tasks with **complete** in the description.

### 3.11. Choosing a threshold for tasks: **threshold**

You can set a comfortable threshold to tell **OOF** when to remind you to complete your tasks.

Format: **threshold HH**

- **time** has to **strictly** be in the format as stated above.

Example:

- **threshold 48**

Example:

```
Enter a command:
threshold 48
Threshold has been updated to 48

Enter a command:
```

Figure 15. Output of *threshold* command

Changes the threshold of the program to 48 hours.

### 3.12. Viewing a summary of a day's task by date: **schedule**

You can view a summary of all the tasks and events on a specific day of your choice.

Format: **schedule DD-MM-YYYY**

- **Date** has to **strictly** be in the format as stated above.

Example:

- **schedule 30-10-2019**

```
Enter a command:
schedule 30-10-2019

Here are your tasks for 30-10-2019:
  1. [E][N] tutorial (from: 30-10-2019 17:00 to: 30-10-2019 18:00)
  2. [D][N] complete lab assignment (by: 30-10-2019 23:59)
  3. [D][N] tutorial (by: 30-10-2019 14:00)
  4. [D][N] lecture (by: 30-10-2019 16:00)

Enter a command:
```

Figure 16. Output of `schedule` command

Provides a summary of a list of todo, deadlines and events that will occur on `30-10-2019`.

### 3.13. Viewing a summary of the next day's task: `summary`

You can view a summary of all the tasks to be done for the next day.

Format: `summary`

Example:

- `summary`

```
Enter a command:
summary

Here are your tasks for 30-10-2019:
  1. [E][N] tutorial (from: 30-10-2019 17:00 to: 30-10-2019 18:00)
  2. [D][N] complete lab assignment (by: 30-10-2019 23:59)
  3. [D][N] tutorial (by: 30-10-2019 14:00)
  4. [D][N] lecture (by: 30-10-2019 16:00)

Enter a command:
```

Figure 17. Output of `summary` command

Provides a summary of a list of todo, deadlines and events that will occur tomorrow.

### 3.14. Viewing free time slots: `free`

You can view the time slots you are available on a specific day so that you can plan project meetings with your friends.

Format: `free DD-MM-YYYY`

- `Date` has to **strictly** be in the format as stated above.

Example:

1. Type `free 30-10-2019` as a command press **ENTER**

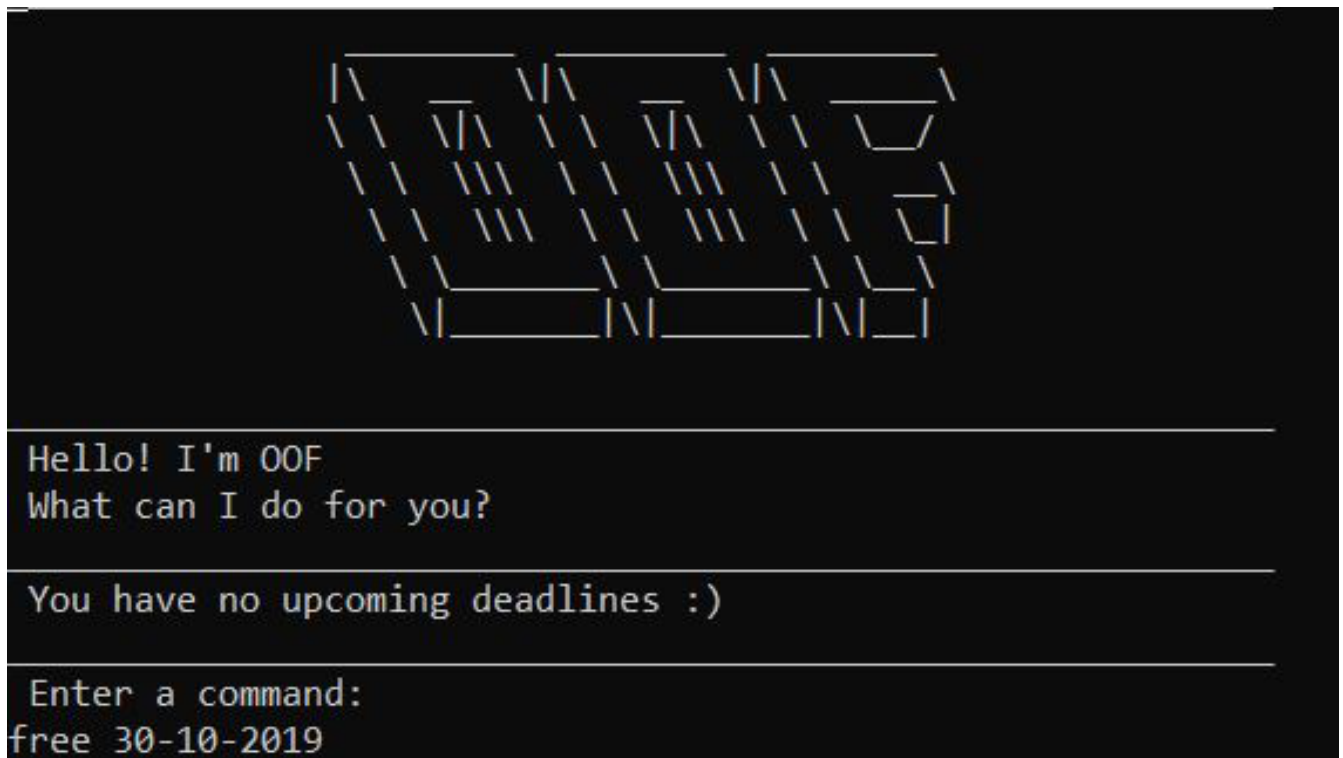


Figure 18. Typing free 30-10-2019 into OOF

2. **00F** displays all the free time slots that you have.



Wednesday 30-10-2019		
07:00 - 08:00	free	
08:00 - 09:00	free	
09:00 - 10:00	free	
10:00 - 11:00	lecture	
11:00 - 12:00	lecture	
12:00 - 13:00	free	
13:00 - 14:00	free	
14:00 - 15:00	free	
15:00 - 16:00	free	
16:00 - 17:00	free	
17:00 - 18:00	tutorial	
18:00 - 19:00	free	
19:00 - 20:00	free	
20:00 - 21:00	free	
21:00 - 22:00	free	
22:00 - 23:00	free	
23:00 - 00:00	free	
Enter a command:		

Figure 19. Typing free with a valid date in the valid format of DD-MM-YYYY



### 3.15. Viewing tasks in week view: `viewweek`

You can view the tasks for any particular week in a table format so that you can have a grasp of what to expect for a particular or even track your own progress.

Format: `viewweek DD MM YYYY`



Note that the parameters `DD MM YYYY` are optional and the command will automatically show tasks for the current week if these parameters are not shown. The tasks for each day are chronologically sorted.

Example:

1. Type `viewweek` as a command and press `ENTER`

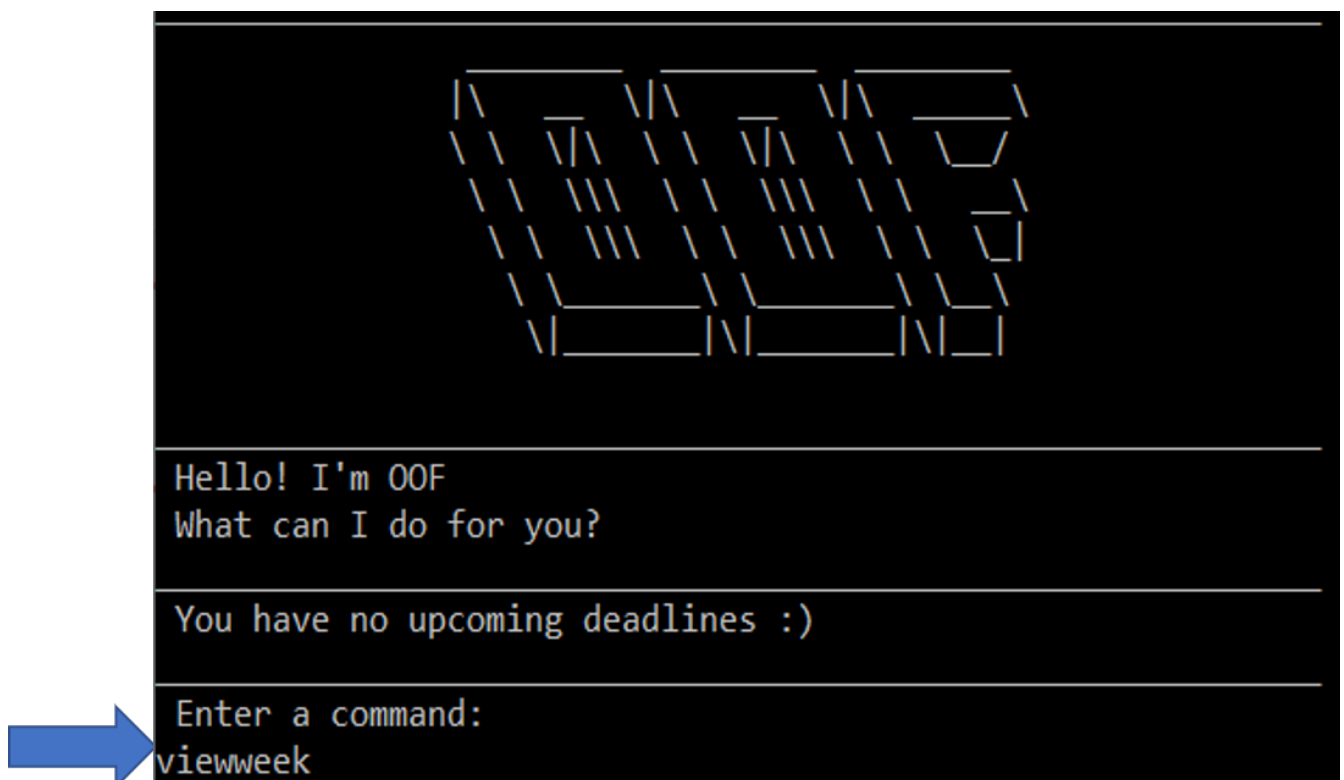


Figure 20. Typing `viewweek` into OOF

2. `OOF` displays the tasks for the week for you.

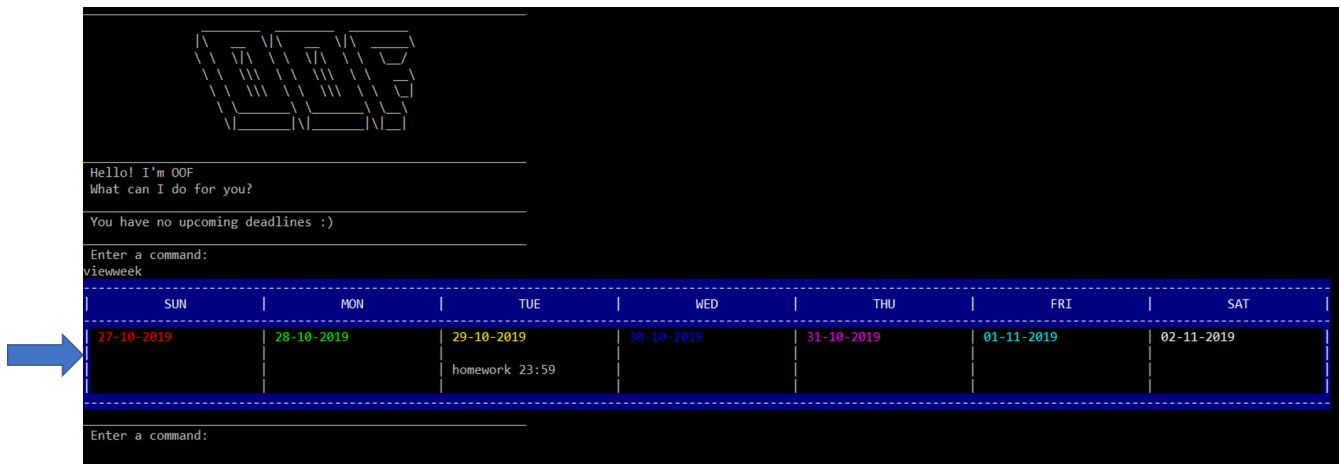


Figure 21. Typing viewweek without date

3. If you wish to display tasks for a particular week, you can input **DD MM YYYY**.

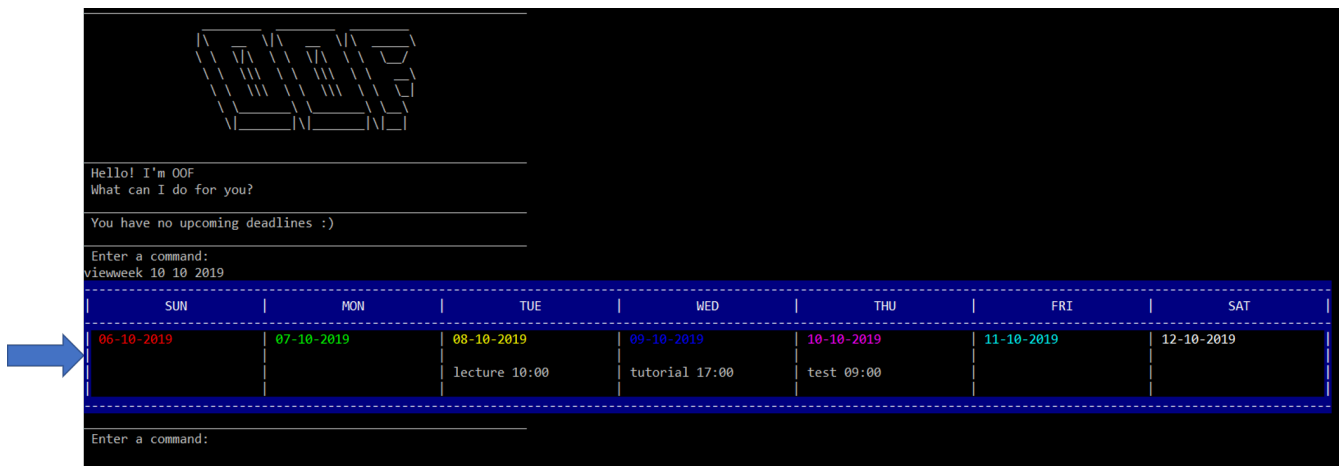


Figure 22. Typing viewweek with date

### 3.16. Viewing all tasks in calendar view: **calendar**

You can view all your tasks for any month so that you are aware of your schedule for that month.

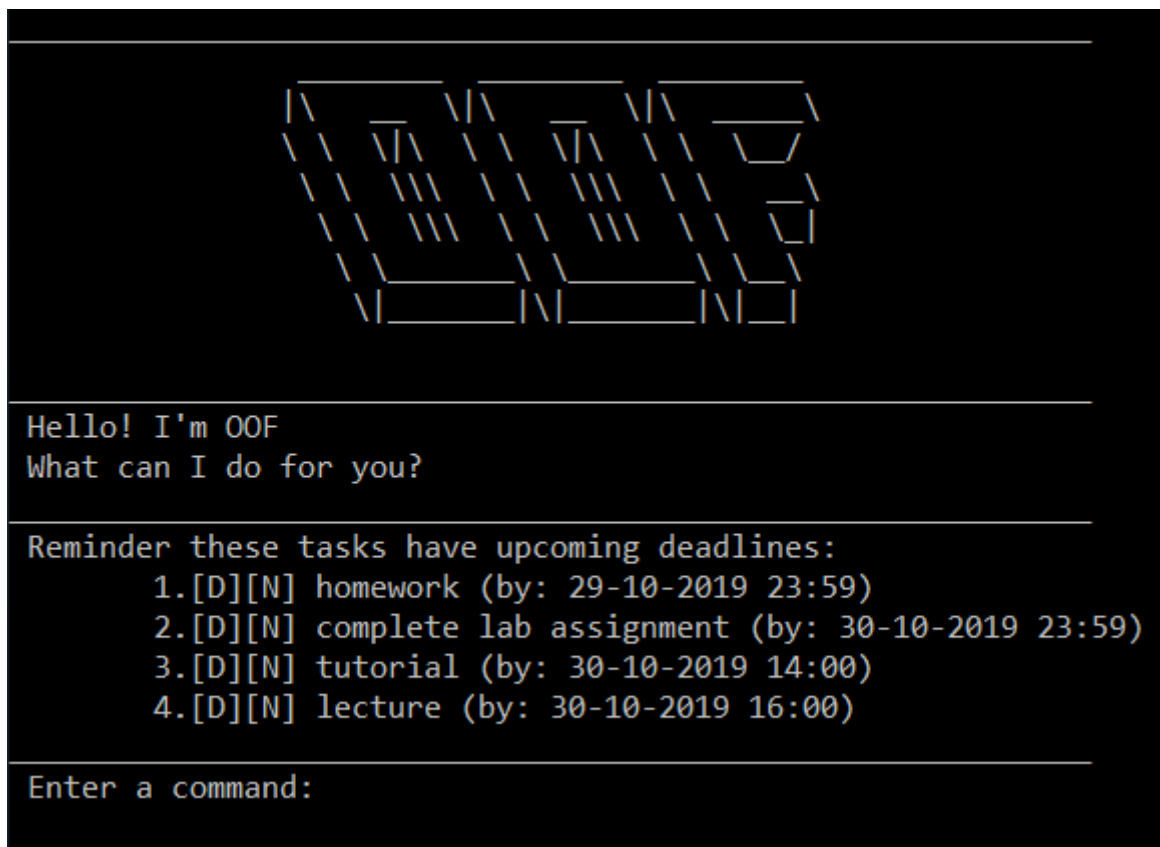
Format: **calendar MM YYYY**

- **MONTH** is an integer from 1-12 (representing January to December).
- **YEAR** is an integer greater than or equal to 0.

Example: **calendar 10 2019**

### 3.17. Setting reminders for upcoming deadlines: **NIL**

You can get timely reminders for the tasks that are expiring.



```
Hello! I'm OOF
What can I do for you?
```

Enter a command:

Figure 23. Output of reminder command

This command functions in the background so **OOF** automatically reminds you of the expiring tasks when you start our program.

### 3.18. Semesters

You can plan ahead for your entire university journey using a few simple commands.

### 3.18.1. Adding semesters: semester /add

You can add a semester to manage your modules.

Format: semester /add YEAR /name SEMESTER /from START\_DATE /to END\_DATE

- **YEAR** represents name of the academic year, **SEMESTER** represents name of the semester, **START\_DATE** and **END\_DATE** represents the start and end date in **dd-MM-yyyy HH:mm** format.

Example:

- semester /add 19/20 /name Semester 2 /from 01-01-2020 /to 05-05-2020

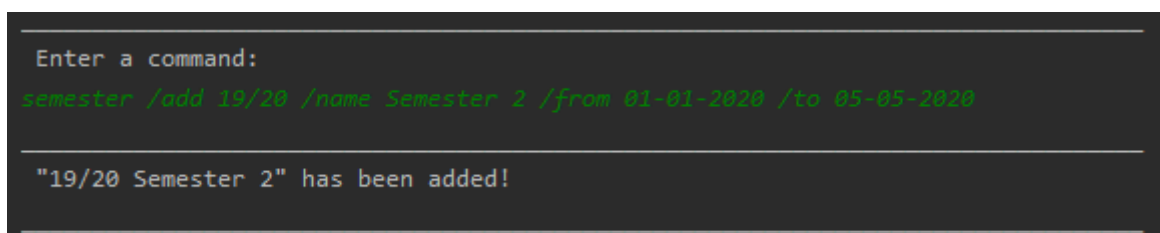


Figure 24. Adding a semester

Adds a semester for Academic Year 19/20, Semester 2 which lasts from 01-01-2020 to 05-05-2020.

### 3.18.2. Viewing semester data: `semester /view`

You can use this command to display all the semesters you have added.

Format: `semester /view`

```
Enter a command:
semester /view

1. Academic Year 19/20, semester 1 (01-10-2019-31-10-2019)
2. Academic Year 19/20, Semester 2 (01-01-2020-05-05-2020)
```

Figure 25. Viewing a semester

### 3.18.3. Removing semester data: `semester /delete`

You can remove unwanted data if you have accidentally added an unwanted semester.

Format: `semester /delete INDEX`

- The **INDEX** refers to the index number displayed in the list of semesters recorded. (`semester /view` can be used to display the added semesters).

Example:

- `semester delete 2`

```
Enter a command:
semester /delete 2

19/20 Semester 2 has been removed.
```

Figure 26. Deleting a semester

Deletes the 2nd semester in the list of semesters.

### 3.18.4. Selecting a semester: `semester /select`

You can select a semester in order to add modules to under a semester.

Format: `semester /select INDEX`

- The **INDEX** refers to the index number displayed in the list of semesters recorded. (`semester /view` can be used to display the added semesters).

Example:

- `semester /select 1`

```
Enter a command:
semester /select 1

"Academic Year 19/20, semester 1 (01-10-2019-31-10-2019)" has been selected!
```

Figure 27. Selecting a semester

Selects the 1st semester in the list of semesters.

## 3.19. Modules

You can keep track of your modules each semester with the help of the module commands.



All commands under modules require a semester to be selected using `semester /select`.

### 3.19.1. Adding module data: `module /add`

You can add a module into `Oof` to manage your lessons and assessments.

Format: `module /add MODULE_CODE /name MODULE_NAME`

- `MODULE_CODE` represents the module code and `MODULE_NAME` represents the module name.

Example:

- `module /add CS2107 /name Introduction to Information Security`

```
Enter a command:
module /add CS2107 /name Introduction to Information Security

"CS2107 Introduction to Information Security" has been added!
```

Figure 28. Adding a module

Adds a module with module code "CS2107" and name as "Introduction to Information Security".

### 3.19.2. Viewing module data: `module /view`

You can display all modules in order to have a quick overview of the modules you are taking this semester.

Format: `module /view`

```
Enter a command:
module /view

Academic Year 19/20, semester 1 (01-10-2019-31-10-2019)
  1. CS2113T software engineering
  2. CS2105 Introduction to Computer Networks
  3. CS2106 Introduction to Operating Systems
  4. CS2107 Introduction to Information Security
```

Figure 29. Viewing a module

### 3.19.3. Removing module data: `module /delete`

You can remove unwanted data if you have accidentally added a wrong module.

Format: `module /delete INDEX`

- The **INDEX** refers to the index number displayed in the list of modules recorded. `module /view` can be used to display the saved semesters).

Example:

- `module /delete 4`

```
Enter a command:
module /delete 4

CS2107 Introduction to Information Security has been removed.
```

Figure 30. Deleting a module

Deletes the 4th module in the list of modules.

### 3.19.4. Selecting a module: `module /select`

You can select a module in order to add lessons for a module.

Format: `module /select INDEX`

- The **INDEX** refers to the index number displayed in the list of modules recorded. (`module /view` can be used to display the added modules).

Example:

- `module /select 3`

```
Enter a command:
module /select 3

"CS2106 Introduction to Operating Systems" has been selected!
```

Figure 31. Selecting a module

Selects the 1st module in the list of modules.

## 3.20. Lessons

Keep track of your lessons for each module with the use of lesson commands!



All command under lesson requires a module to be selected using `module /select`.

### 3.20.1. Viewing lesson data: `lesson`

You can display all the lessons you have added if you wish to view all lessons for a module.

Format: `lesson`

```
Enter a command:
lesson

CS2106 Introduction to Operating Systems
  1. tutorial, MONDAY 16:00 to 18:00
```

Figure 32. Viewing list of modules

### 3.20.2. Adding lesson data: `lesson /add`

You can add a lesson into `Oof`.

Format: `lesson /add NAME /day DAY /from START_TIME /to END_TIME`

- `NAME` of the lesson can have multiple words, not just limited to single-word descriptions.
- `DAY` of the lesson ranges from `MONDAY` to `SUNDAY`.
- `START_TIME` and `END_TIME` have to **strictly** be in the `HH:MM` format.

Example:

- `lesson /add lecture /day WEDNESDAY /from 14:00 /to 16:00`  
Adds a lecture on Wednesday from 14:00 to 16:00 for the selected module.

```
Enter a command:
lesson /add lecture /day WEDNESDAY /from 14:00 /to 16:00

"CS2106 lecture" has been added!
```

Figure 33. Adding a lesson

### 3.20.3. Removing lesson data: `lesson /delete`

You can remove unwanted data if you have added the wrong date for a lesson.

Format: `lesson /delete INDEX`

- The **INDEX** refers to the index number displayed in the list of lessons recorded. `lesson /view` can be used to display the saved lessons).

Example:

- `lesson /delete 1`

```
Enter a command:
lesson /delete 1

CS2106 tutorial has been removed.
```

Figure 34. Deleting a lesson

Deletes the 1st lesson in the list of lessons.

## 3.21. Adding assessment data: `assessment`

You can keep track of assessments by adding assessments.

Format: `assessment DESCRIPTION /from DD-MM-YYYY HH:MM /to DD-MM-YYYY HH:MM`



Requires a module to be selected using `module /select`.

- **Description** of the assessment can have multiple words, not just limited to single-word descriptions.
- **Date and time** have to **strictly** be in the format as stated above.

Example:

- `assessment final examination /from 31-10-2019 16:00 /to 31-10-2019 18:00`

Adds an assessment with name, start and end time as `final examination`, `25-11-2019 13:00`, `25-11-2019 15:00` respectively.



```
Enter a command:
assessment final examination /from 31-10-2019 16:00 /to 31-10-2019 18:00

Got it. I've added this task:
[A][N] CS2106 final examination (from: 31-10-2019 16:00 to: 31-10-2019 18:00)
Now you have 20 tasks in your list.
```

Figure 35. Adding an assessment

## 3.22. Adding assignment data: **assignment**

You can use this command to keep track of an assignment for a particular module.

Format: **assignment** DESCRIPTION /by DD-MM-YYYY HH:MM



Requires a module to be selected using **module** /select.

- **Description** of the assessment can have multiple words, not just limited to single-word descriptions.
- **Date and time** have to **strictly** be in the format as stated above.

Example:

- **assignment** lab /by 23-11-2019 23:59

```
Enter a command:
assignment lab /by 23-11-2019 23:59

Got it. I've added this task:
[A][N] CS2106 lab (by: 23-11-2019 23:59)
Now you have 21 tasks in your list.
```

Figure 36. Adding an assignment

Adds an assignment **lab** for the selected module with the due date as **23-11-2019 23:59**.

## 3.23. Start Assignment Tracker: **start**

You can start your tracking of an assignment from the current time.

Format: **start** MODULE\_CODE ASSIGNMENT\_DESCRIPTION

Example: **start** cs2113t user guide

```
Enter a command:
start cs2113t user guide

Begin Assignment: cs2113t user guide
It is currently Tue Oct 29 18:37:31 SGT 2019
Current total time spent on user guide: 100 minutes
```

Figure 37. Output of StartTracker command

## 3.24. Pause Assignment Tracker: `pause`

You can pause your tracking of an assignment at the current time.

Format: `pause MODULE_CODE ASSIGNMENT_DESCRIPTION`

Example: `pause cs2113t user guide`

```
Enter a command:
pause cs2113t user guide

Pausing Assignment: cs2113t user guide
It is currently Tue Oct 29 18:41:21 SGT 2019
Total time spent on user guide: 104 minutes
```

Figure 38. Output of PauseTracker command

## 3.25. Stop Assignment Tracker: `stop`

You can stop your tracking of an assignment at the current time with `stop`.

Format: `stop MODULE_CODE ASSIGNMENT_DESCRIPTION`

Example: `stop cs2113t user guide`

```
Enter a command:
stop cs2113t user guide

Ending Assignment: cs2113t user guide
It is currently Tue Oct 29 18:41:49 SGT 2019
Total time spent on user guide: 104 minutes
```

Figure 39. Output of StopTracker command

## 3.26. View Assignment Tracker: `viewTracker`

You can view a histogram featuring the amount of time you spend on each module in blocks of 10 minutes with `viewtracker`.

Format: `viewTracker`

```
Enter a command:
viewtracker

|
| # cs2101 — 12 minutes
|
| #### cs2105 — 40 minutes
|
| ##### cs2113t — 100 minutes
```

Figure 40. Output of ViewTracker command

Format: `viewTracker TIME_PERIOD` [coming soon in v1.4]

### Options for TIME\_PERIOD

#### Day

filter time spent on each `Module` today

#### Week

filter time spent on each `Module` over the last 7 days

Example: `viewTracker Day`

## 3.27. Exiting the program: `bye`

Exits the program.

Format: `bye`

## 3.28. View undone tasks brought forward to the next day: `undone` [coming soon in v2.0]

You can view the list of all the tasks not done that were brought forward to the next day.

Format: `undone`

Example:

- `undone` You can postpone the tasks that were not fulfilled to the next day.

## 3.29. Filter tasks by categories: `filter` [coming soon in v2.0]

You can filter tasks by matching the category given.

Format: `filter CATEGORY`

- `Category` of the task can be any one of the following: todo, deadline, event, recurring.

Example:

- `filter todo`  
You can display all todo tasks.

### 3.30. Adding a task: `tentative` [coming soon in v2.0]

You can add a task that can be confirmed at a later time.

Format: `tentative DESCRIPTION`

- `Description` of the task to be done can have multiple words, not just limited to single-word descriptions.

Example:

- `tentative group lunch`  
Adds a tentative task called `group lunch`.

### 3.31. Adding a task: `do-after` [coming soon in v2.0]

You can add a task that needs to be done after a specified task.

Format: `do-after INDEX DESCRIPTION`

- The `INDEX` refers to the index number displayed in the list of tasks recorded. (`list` can be used to display the saved tasks).
- `Description` of the task to be done can have multiple words, not just limited to single-word descriptions.

Example:

- `do-after 1 buy groceries`  
Adds a do-after task called `buy groceries` that will be displayed once the 1st task in the list has been completed.

### 3.32. Add estimated time taken: `estimate` [coming soon in v2.0]

You can add the estimated time taken to complete a task.

Format: `estimate INDEX HH`

- The `INDEX` refers to the index number displayed in the list of tasks recorded. (`list` can be used to display the saved tasks).
- `time` has to **strictly** be in the format as stated above.

Example:

- `estimate 1 48`

Adds to the 1st task the estimated time taken of 48 hours to complete it.

### 3.33. Adding a task: `range` [coming soon in v2.0]

You can add a task that needs to be completed within a certain time period

Format: `range DESCRIPTION /from DD-MM-YYYY HH:MM /to DD-MM-YYYY HH:MM`

- **Description** of the task to be done can have multiple words, not just limited to single-word descriptions.
- **Date and time** have to **strictly** be in the format as stated above.

Example:

- `range study for exam /from 01-10-2019 21:00 /to 05-10-2019 11:00`  
Adds a task with description and time period to `study for exam` and between `01-10-2019 21:00` to `05-10-2019 11:00`.

### 3.34. View two different calendars side-by-side: `viewDual` [coming soon in v2.0]

Transforms all current tasks into two calendar views, one for tutor tasks and one for student tasks.

Format: `viewDual`

### 3.35. Export calendar: `export` [coming soon in v2.0]

You can export all current tasks recorded into a shareable format in calendar view.

Format: `export`

## 4. FAQ

**Q:** How do I view my tasks on the Calendar?

**A:** You can use the `calendar` command.

**Q:** How do I transfer my data to another Computer?

**A:** You can copy the entire directory containing our program into the destination directory.

**Q:** How do I save my tasks in **OOF**?

**A:** You are not needed to explicitly save the tasks as **OOF** will automatically save all tasks that are added during runtime.

## 5. Command Summary

## 5.1. Available Commands

View the list of features and their usages.

- **Help:** `help`



You can view the usage of a specific command by typing `help COMMAND`, where `COMMAND` is the name of the feature. e.g. `help calendar`

Add a task with a deadline.

- **Deadline:** `deadline DESCRIPTION /by DD-MM-YYYY HH:MM`  
e.g. `deadline homework /by 20-09-2019 13:00`

Add an event with start and end time.

- **Event:** `event DESCRIPTION /from DD-MM-YYYY HH:MM /to DD-MM-YYYY HH:MM`  
e.g. `event project meeting /from 20-09-2019 13:00 /to 20-09-2019 17:00`

Add a todo with a specific date.

- **Todo:** `todo DESCRIPTION /on DD-MM-YYYY`  
e.g. `todo withdraw money /on 19-09-2019`

Set a recurring task.

- **Recurring:** `recurring INDEX NUMBER_OF_OCCURRENCES`  
e.g. `recurring 4 3`



You will be prompted to enter a number from 1-4 afterward.

1 represents `DAILY`.  
2 represents `WEEKLY`.  
3 represents `MONTHLY`.  
4 represents `YEARLY`.

List all the task you have saved in **OOF**

- **List:** `list`

Mark a task as done.

- **Done:** `done INDEX`  
e.g. `done 1`

Delete a specific task.

- **Delete:** `delete INDEX`  
e.g. `delete 1`

Find anything using keywords.

- **Find:** `find DESCRIPTION`  
e.g. `find withdraw money`

Set a threshold in hours for reminders.

- **Threshold:** `threshold HH`  
e.g. `threshold 48`

Check your schedule on a particular day.

- **Schedule:** `schedule DD-MM-YYYY`  
e.g. `schedule 04-10-2019`

View a summary of your tasks for the next day.

- **Summary:** `summary`

View free time slots on a specific day.

- **Free:** `free DD-MM-YYYY`  
e.g. `free 10-10-2019`

View all tasks in a table form for any particular week.

- **ViewWeek:** `viewweek DD MM YYYY` e.g. `viewweek 30 10 2019`



Note that the parameters `DD MM YYYY` are optional and the command will automatically show tasks for the current week if these parameters are not shown. The tasks in each day is chronologically sorted.

View tasks for any particular month in calendar format.

- **Calendar:** `calendar MM YYYY`  
e.g. `calendar 10 2019`



Note that the parameters `DD MM YYYY` are optional and the command will automatically show tasks for the current month if these parameters are not shown. The tasks in each day is chronologically sorted.

View reminder based on the threshold set.

- **Reminder:** `NIL`



This feature runs in the background thus no input is needed from you.

Starts assignment tracker.

- **Start Assignment Tracker:** `start MODULE_CODE ASSIGNMENT_DESCRIPTION`  
e.g. `start cs2113t user guide`

Pauses assignment tracker.

- **Pause Assignment Tracker:** `pause MODULE_CODE ASSIGNMENT_DESCRIPTION`  
e.g. `pause cs2113t user guide`

Stops assignment tracker.

- **Stop Assignment Tracker:** `stop MODULE_CODE ASSIGNMENT_DESCRIPTION`  
e.g. `stop cs2113t user guide`

View assignment tracker diagram.

- **View Assignment Tracker:** `viewtracker`  
e.g. `viewtracker`

Exit OOF by using this command.

- **Bye:** `bye`

## 5.2. Coming Soon

- **Tentative:** `tentative DESCRIPTION`
- **Do-after:** `Do-after INDEX DESCRIPTION`
- **Filter:** `filter CATEGORY`
- **ViewUndone:** `viewUndone`
- **Estimate:** `estimate`
- **Range:** `range`
- **ViewDual:** `viewDual`
- **Export:** `export`