

INTRO TO SOFTWARE ENGINEERING (CS-251)

Code Report



Assignment 1 Task 1

Name	ID	Hours spent learning java
Yaseen Mohamed Kamal	20230468	~ 20



Todo – List

Click [here](#), for Demonstration Video

Description

The program is a simple Command line-Interface Todo list with many basic and intermediate-advanced features.

We will go through a feature list then an explanation of the main function flow

Feature Name	Brief Explanation	Number
Create new task	Adds a new task to the list, with basic information supplied like name, deadline and description	1
Remove existing task	Removes existing task supplied index num	2
Display all tasks	Prints out all saved tasks in a table-like form	3
Edit existing tasks	Change task deadline or description	4
Mark as done	Flags task as finished and updates statistic	5
Change list order	Automatically sorts by deadline or Importance ascendingly	6
Change task priority	Changes default task priority value of 1	7
View task statistics	Displays Finished and unfinished tasks	8
Clear all	Deletes all tasks	9
Undo last delete	Brings back last removed task	10
Exit	Exits program	11

Main function flow

Start by initializing a scanner object to take input, going to a while loop menu with break case only when input is 11, displaying menu, and taking input in choice variable, and going through a switch-case statement to decide which operation to conduct



Screenshots:

```
File Edit Selection View ... A1-T1-Yaseen
Main.java X todo.java item.java 2
Main.java > Main > main(String[])
9
10
11 public class Main {
12
13     void Main() {
14     }
15     Run[Debug]
16     public static void main(String[] args) {
17         int choice=0;
18         Scanner reader = new Scanner(System.in);
19
20         System.out.println(x:"Welcome to TODO List");
21         TODO newlist = new TODO();
22         while(choice != 11)
23         {
24             System.out.println(x:"\nPlease select from the menu below: ");
25             System.out.println(x:"1. Create a new task\n2. Remove existing task\n3. Display all tasks\n4. Edit existing
26             System.out.printf(format:"9. Clear all\n10. Undo last delete\n11. Exit\n-> ");
27             choice = reader.nextInt();
28             reader.nextLine();
29
30             switch(choice)
31             {
32                 case 1: //creating a new task
33                     String taskname;
34                     System.out.printf(format:"Please enter taskname: ");
35                     taskname = reader.nextLine();
36                 }
37             }
38         }
39     }
40 }
Ln 20, Col 32 (4 selected) Spaces: 4 UTF-8 LF (1) Java
```

```
File Edit Selection View ... A1-T1-Yaseen
Main.java X todo.java item.java 2
Main.java > Main > main(String[])
11 public class Main {
12     public static void main(String[] args) {
13
14         switch(choice)
15         {
16             case 1: //creating a new task
17                 String taskname;
18                 System.out.printf(format:"Please enter taskname: ");
19                 taskname = reader.nextLine();
20
21                 int deadline;
22                 System.out.printf(format:"deadline[HHmm]: ");
23                 while(true)
24                 {
25                     deadline = reader.nextInt();
26                     reader.nextLine();
27                     if(deadline < 0 || deadline > 2359)
28                     {
29                         System.out.println(x:"Error: deadline has to be between 0000 and 2359, please try again...");
30                         System.out.printf(format:"deadline[HHmm]: ");
31                     }
32                     else{
33                         break;
34                     }
35                 }
36             }
37         }
38     }
39 }
Ln 20, Col 32 (4 selected) Spaces: 4 UTF-8 LF (1) Java
```

```
File Edit Selection View ... A1-T1-Yaseen
Main.java X todo.java item.java 2
Main.java > Main > main(String[])
11 public class Main {
12     public static void main(String[] args) {
13         newlist = newlist;
14         //System.out.println(newlist); for debugging TODO remove at the end
15         break;
16
17         case 2:
18             int index;
19             System.out.printf(format:"Please enter task index to remove: ");
20             index = reader.nextInt();
21             newlist.remove(index);
22             newlist.remove(index);
23             break;
24
25         case 3:
26             newlist.displayallTasks();
27             break;
28
29         case 4:
30             System.out.println(x:"which field would you like to change?[(T)ime, (D)escription]");
31             System.out.print(s:"-> ");
32             String Choice;
33             Choice = reader.nextLine();
34             int task;
35             System.out.println(x:"which task would you like to edit?");
36             System.out.print(s:"-> ");
37             task = reader.nextInt();
38             newlist.editTask(Choice, task);
39             break;
40
41         case 5:
42             break;
43     }
44 }
Ln 20, Col 32 (4 selected) Spaces: 4 UTF-8 LF (1) Java
```



```
File Edit Selection View ... A1-T1-Yaseen
Main.java 1 todo.java X item.java 2
todo.java > todo > Items
1 public class todo extends Item { //this class is for the list
2     protected int nItems=0;
3     ArrayList<Item> items;
4     protected int[] stats = new int[2]; //index 0: for finished, index 1: for unfinished
5     item lastdeleted;
6
7     public todo() //create a new List instance
8     {
9         items = new ArrayList<>();
10    }
11
12    void addItem(String iName, int iTime, String iDesc, int Importance)
13    {
14        Importance = 1;
15        //iDesc = ""; //was a failed attempt to try and initialize the variable
16        if(iDesc == "")
17        {
18            item newItem = new item(iName, iTime, Importance);
19            items.add(newItem);
20            newItem.displayItem();
21        }
22        else
23        {
24            item newItem = new item(iName, iTime, iDesc, Importance);
25            items.add(newItem);
26        }
27    }
28 }
```

```
File Edit Selection View ... A1-T1-Yaseen
Main.java 1 todo.java X item.java 2
todo.java > todo > updateStatus()
1 public class todo extends Item { //this class is for the list
2
3     void removeItem(int index)
4     {
5         lastdeleted = items.get(index);
6         items.remove(index);
7         nItems--;
8     }
9
10    void displayAllTasks() {
11        System.out.println("Index\tName\tDeadline\tDescription\tPriority\tStatus");
12        for (int i = 0; i < nItems; i++) {
13            item currentItem = items.get(i);
14            String status = currentItem.status ? "finished" : "pending";
15            System.out.println(String.format("%d\t%s\t%d\t%s\t%d\t%s",
16                i, currentItem.name, currentItem.time, currentItem.description, currentItem.importance, status));
17        }
18    }
19
20    void editTask(String usin, int index)
21    {
22        items.get(index).editItem(usin);
23    }
24
25    boolean finishTask(int ind)
26 }
```

```
File Edit Selection View ... A1-T1-Yaseen
Main.java 1 todo.java X item.java 2 X
item.java > item > importance
1 import java.util.Scanner;
2
3 public class item {
4     public String name; //item name
5     public int time; //item scheduled time
6     public String description = ""; //item description, may or may not exist
7     public boolean status = false; //item status, true when done
8     public int importance; //from 1 to 3, dictating level of importance
9
10    public item()
11    {
12    }
13
14    public item(String name, int time, String description, int importance) //create a new item when description exists
15    {
16        this.name = name;
17        this.time = time;
18        this.description = description;
19        this.importance = importance;
20    }
21
22    public item(String name, int time, int importance) //create new item without description
23    {
24        this.name = name;
25        this.time = time;
26        this.importance = importance;
27    }
28 }
```

A screenshot of an IDE window titled 'AI-Ti-Yaseen'. The editor shows a Java file named 'item.java' with the following code:

```
39
40
41 public void displayItems()    //special implementation for when displaying everything
42 {
43     int descLength = this.description.length();
44     System.out.print( this.name + "\t\t" + this.time + "\t\t\t"); // this.description + "\t\t" + this.Importance
45     /* for(int i = 0; i < descLength; ++i)
46     {
47         System.out.print(" ");
48     }*/
49
50     System.out.print(this.description + "\t\t" + this.Importance+ "\t");
51
52     if(!status)
53         System.out.println(x:"pending");
54     else
55         System.out.println(x:"finished");
56 }
57
58 public void editItem(String choice)
59 {
60     Scanner classer = new Scanner(System.in);
61     if(choice.equals(anObject:"T") || choice.equals(anObject:"Time"))    //change deadline time
62     {
63         int nTime;
```

The IDE interface includes a menu bar (File, Edit, Selection, View, ...), a toolbar, and a sidebar with icons for Explorer, Search, Run and Debug, and Extensions. The status bar at the bottom shows 'Ln 8, Col 50 (6 selected)', 'Spaces: 4', 'UTF-8', 'CRLF', and '(1) Java'.