GROUP ASSIGNMENT

TECHNOLOGY PARK MALAYSIA

CT042-3-1-IDB

INTRODUCTION TO C

HAND OUT DATE: 8 May 2021

HAND IN DATE: 22 June 2022

WEIGHTAGE: 50%

INSTRUCTIONS TO CANDIDATES:

1. **Submit your assignment at the administrative counter.**
2. **Students are advised to underpin their answers with the use of references (cited using the Harvard Name System of Referencing).**
3. **Late submission will be awarded zero (0) unless Extenuating Circumstances (EC) are upheld.**
4. **Cases of plagiarism will be penalized.**
5. **The assignment should be bound in an appropriate style (comb bound or stapled).**
6. **Where the assignment should be submitted in both hardcopy and softcopy, the softcopy of the written assignment and source code (where appropriate) should be on a CD in an envelope / CD cover and attached to the hardcopy.**
7. **You must obtain 50% overall to pass this module.**

Contents

Figures 2

Introduction 3
Assumptions 3
Design of the program – Psuedo codes 4
To do list functions 4
Create my day schedule functions 8
My day functions 10
Miscellaneous functions 13
Main 18
Design of the code – Flowcharts 19
Main flow 19
Todolist function 20
My day function 23
Additional features 25
Screenshots of code 25
Main menu 25
To do list option 25
Create my day Schedule 30
My day 37
Test plan 38
Main menu 38
To do list option 39
Create my day schedule 44
My day 48
Conclusion 49
References 50

Figures

Figure 1 25
Figure 2 26
Figure 3 26
Figure 4 27
Figure 5 28

Figure 6 29
Figure 7 30
Figure 8 31
Figure 9 31
Figure 10 32
Figure 11 32
Figure 12 33
Figure 13 33
Figure 14 34
Figure 16 35
Figure 17 36
Figure 18 36
Figure 19 37
Figure 20 37
Figure 21 38

Introduction

This documentation is for a time management application which helps the user save their time by time blocking their day. Users can create and maintain to do lists, tasks can be added, updated and deleted in these to do lists.

Assumptions

Add_task function:

- Tasks have a task description, duration, due date, category and date created attributes
- Date created cannot be entered by the user but is generated by the system
- Tasks duration can only be 0,15,45 or 60 minutes

Display_tasks function

- A line in the to do list can only be 200 characters long at most

Update_task function

- User cannot update the date created attribute of a task

todolist function

- Only 4 commands can be used namely: add, del, upd and srt

Create_time_blocks function

- Time blocks can be of 2 types: either rest or work
- Each time block is 1 hour

Choose_tasks_myday function

- Only tasks from to do list can be chosen for my day list

Map_tasks_to_timeblocks function

- Only work time blocks can be allocated tasks
- One time block cannot have tasks that have a total duration which exceeds one hour

Update_my_day_task_progress function

- Task progress is a value from 0 to 100
- Tasks that are completed must be removed from my day list

Compare_dates, selection_sort_date and validate_date functions

- Date is in the form dd/mm/yyyy

Design of the program – Psuedo codes

To do list functions

```
DEFINE add_task(file_url,user_input[])
    DECLARE file
    file=OPEN FILE(file_url,a)
    DECLARE date
    date=generate_date()

    IF validate_date(user_input[3])==-1 THEN
        RETURN -1
    ENDIF
    WRITE TO FILE(file,user_input[1],user_input[2],user_input[3],user_input[4],date)
    CLOSE FILE(file)
ENDDEFINE
```

```
DEFINE update_task(file_url[],line_pos)
    DECLARE attribute_pos,row[],new_value[],file,temp
    file = OPEN FILE(file_url,r)
    temp = OPEN FILE("temp.txt",w+)

    DISPLAY "Which attribute do you want to update?\n1.Task description\n2.Duration \n3.Due date\n4.Category\n"
    READ INPUT(attribute_pos)

    IF attribute_pos==1 THEN
        RETURN -1
    ENDIF

    IF attribute_pos!=1 && attribute_pos!=2 &&  attribute_pos!=3 && attribute_pos!=4 THEN
        DISPLAY "Error: Invalid option\n"
        RETURN -1
    ENDIF

    DISPLAY "What is the new value?\n"
    READ INPUT(new_value)
    attribute_pos--
    IF attribute_pos==1 THEN
        IF validate_duration(new_value)==-1 THEN
            DISPLAY "this is the one being taken\n"
            RETURN -1
        ENDIF
    ENDIF
    IF attribute_pos==2 THEN
        IF validate_date(new_value)==-1 THEN
            RETURN -1
        ENDIF
    ENDIF
    LOOP (i=0;!END OF FILE;i++)
        READ FILE LINE(file,row)
        IF i==line_pos THEN
            DECLARE line_to_edit[][]
            READ FROM STRING(row,line_to_edit[0],line_to_edit[1],line_to_edit[2],line_to_edit[3],line_to_edit[4]) IN LOOPMAT "%s %s %s %s %s"
            line_to_edit[attribute_pos]=new_value
            WRITE TO FILE(temp,line_to_edit[0],line_to_edit[1],line_to_edit[2],line_to_edit[3],line_to_edit[4])
        ENDIF
        ELSE
            IF (END OF FILE) THEN
                 BREAK
            ENDIF
            WRITE TO FILE(temp,row)
        ENDELSE
    ENDLOOP

    CLOSE FILE(file)
    CLOSE FILE(temp)
    REMOVE(file_url)
    RENAME("temp.txt",file_url)

    RETURN 0
ENDDEFINE
```

```
DEFINE delete_task(line_position, file_url[])

    DECLARE file,temp,row
    file = OPEN FILE(file_url,r)
    temp = OPEN FILE("temp.txt",w+)
    LOOP(i=0;!END OF FILE(file);i++)
        READ FILE(file,row)
        IF(i==line_position) THEN
            CONTINUE
        ENDIF
        IF(END OF FILE(file))THEN
            BREAK
        ENDIF

        WRITE TO FILE(temp,row)
    ENDLOOP

    CLOSE FILE(file)
    CLOSE FILE(temp)

    //rename temp.txt to todolist.txt
    DECLARE ret
    REMOVE("todolist.txt")
    RENAME("temp.txt","todolist.txt");
ENDDEFINE
```

```
DEFINE sort_list(file_url)

    DECLARE row[200],file,tasks[][][],i
    file = OPEN FILE(file_url,r)
    LOOP(i=0;!END OF FILE(file);i++)
        READ FILE LINE(file,row)
        IF(row==""" || i==0) THEN
            CONTINUE
        ENDIF
        READ FROM STRING(row,tasks[i-1][0],tasks[i-1][1],tasks[i-1][2],tasks[i-1][3],tasks[i-1][4]) IN FORMAT "%s %s %s %s %s"
        IF (END OF FILE(file))THEN
                BREAK
        ENDIF
    ENDLOOP
    CLOSE FILE(file)
    DECLARE num_tasks
    num_tasks = i

    DISPLAY "Sort by:\n1.Task description\n2.Duration\n3.Due date\n4.Category\n5.Date created\n"
    DECLARE user_input
    user_input=0
    READ INPUT(user_input)
    DISPLAY "What order? Ascending=1;Descending=0;\n"
    DECLARE asc
    READ INPUT(asc)


    IF(user_input==1 || user_input==2 || user_input==4)THEN
        selectionSort(tasks,num_tasks,user_input-1,asc);
    ENDIF
    ELSEIF(user_input==3 || user_input==5) THEN
        selectionSortDate(tasks,num_tasks,user_input-1,asc);
    ENDIF
ENDEFINE
```

```
DEFINE todolist(todolist_file_url)
    DECLARE todolist,c
    todolist = OPEN FILE(todolist_file_url,r)
    c = GET FILE CHAR(todolist)
    c = GET FILE CHAR(todolist)
    CLOSE FILE(todolist)

    IF c=='\n' THEN
        DISPLAY "[Blank file: please add some tasks]\n"
    ENDIF

    DISPLAY "\nYour TO DO list:\n"
    DECLARE num_tasks
    num_tasks = display_tasks("todolist.txt");

    DECLARE command[]
    READ INPUT(command)
    DECLARE command_kword[]
    command_kword = [command[0],command[1],command[2],'\0']

    IF(strcmp(command_kword,"add")==0) THEN
        DECLARE words[][]
        READ STRING(command,words[0],words[1],words[2],words[3],words[4],words[5])IN LOOPMAT "%s %s %s %s %s %s"

        IF(words[2]!="0" && words[2]!="15" && words[2]!="30" && words[2]!="45" && words[2]!="60") THEN
            DISPLAY "Error: Task duration can only be 0,15,30,45 or 60 minutes\n"
        ENDIF
        ELSE{
            add_task("todolist.txt",words)
        ENDELSE

    ENDIF
    IF command_kword=="del" THEN
        DECLARE words[][]
        READ FROM STRING(command,words[0],words[1])IN LOOPMAT "%s %s"

        DECLARE line_to_del
        line_to_del = atoi(words[1]);

        IF(line_to_del<=0) THEN
            DISPLAY "Error: Invalid task position\n"
        ENDIF

        ELSEIF(line_to_del>num_tasks) THEN
            DISPLAY "Error:Task position does not exist\n"
        ENDELSEIF
        ELSE
            delete_task(line_to_del,todolist_file_url)
        ENDELSE
    ENDIF
```

(CONTINUED IN NEXT SCREENSHOT)

```
    IF command_kword=="upd" THEN
        DECLARE words[2][30]
        READ FROM STRING(command,words[0],words[1]) IN LOOPMAT "%s %s"
       DECLARE line_to_update
       line_to_update = atoi(words[1])

        IF line_to_update<=0 THEN
            DISPLAY "Error: Invalid task position\n"
        ENDIF
        ELSEIF(line_to_update>num_tasks)THEN
            DISPLAY "Error:Task position does not exist\n"
        ENDIF
        ELSE
            update_task("todolist.txt",line_to_update)
        ENDELSE
    ENDIF
    IF command_kword=="srt" THEN
        sort_list("todolist.txt")
    ENDIF
ENDDEFINE
```

Create my day schedule functions

```
DEFINE create_time_blocks(){
    DECLARE time_blocks[]
    time_blocks = [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
    DECLARE time_intervals[]
    time_intervals = ["6:00 - 7:00","7:00 - 8:00","8:00 - 9:00","9:00 - 10:00","10:00 - 11:00","11:00 - 12:00","12:00 - 13:00","13:00 - 14:00","14:00 - 15:00","15:00 - 16:00","16:00 - 17:00","17:00 - 18:00","18:00 - 19:00","19:00 - 20:00","20:00 - 21:00",
    "21:00 - 22:00","22:00 - 23:00","23:00 - 24:00","24:00 - 1:00","1:00 - 2:00","2:00 - 3:00","3:00 - 4:00","4:00 - 5:00","5:00 - 6:00"]
    DOWHILE(1){
        DECLARE time_block_index,RW
        DISPLAY "\Step 1. Time blocks in a day\n"
        LOOP(int i=0;i<24;i++)
            IF time_blocks[i]==0 THEN
                DISPLAY "%d %s %s \n",i+1,time_intervals[i],"Rest"
            ENDIF
            ELSE
                DISPLAY i+1,time_intervals[i],"Work"
            ENDELSE
        ENDLOOP
        DISPLAY "Select time block (press 101 when you are done): \n"
        READ INPUT (time_block_index)

        IF time_block_index == 101 THEN
            BREAK
        ENDIF

        IF time_block_index<0 || time_block_index>24 THEN
            DISPLAY "Error:Invalid input\n"
            return RETURN 1
        ENDIF

        DISPLAY "Is the time rest or work? R/W\n"
        READ INPUT(RW)

        IF RW =='R' | RW == 'r' THEN
                time_blocks[time_block_index-1]=0;
        ENDIF
        ELSE
                time_blocks[time_block_index-1]=1;
        ENDELSE
    ENDWHILE

    DECLARE work_hours_count
    work_hours_count = 0
    LOOP(int i=0;i<24;i++)
        IF time_blocks[i]==0 THEN
            DISPLAY i+1,time_intervals[i],"Rest"
        ENDIF
        ELSE
            DISPLAY i+1,time_intervals[i],"Work"
            work_hours_count++
        ENDELSE
    ENDLOOP
    DISPLAY "Your workhours LOOP the day: %d hours\n",work_hours_count

    RETURN time_blocks;
ENDDEFINE
```

```
DEFINE choose_tasks_myday(char file_url[20]){

    DECLARE my_day_task_pos[20]
    DISPLAY "\nTO DO list\n"
    DECLARE num_tasks
    num_tasks = display_tasks(file_url)

    DISPLAY "Step2. Choose task(101 to exit)\n"
    LOOP(i=0;i<20;i++)
        scanf("%d",&my_day_task_pos[i]);
        READ INPUT(my_day_task_pos[i])
        IF my_day_task_pos[i]==101 THEN
            my_day_task_pos[i]=0;
            BREAK
        ENDIF
        IF my_day_task_pos[i]<=0 || my_day_task_pos[i]>num_tasks THEN
            DISPLAY "Error:invalid task "
            RETURN -1
        ENDIF
    ENDLOOP
    RETURN my_day_task_pos
ENDDEFINE
```

```
DEFINE display_work_time_blocks(time_block_array, time_intervals[])
    DISPLAY "Time blocks:\n"
    DECLARE pos
    pos=0
    LOOP i=0;i<24;i++
       IF time_block_array[i]==1 THEN
           DISPLAY ++pos,time_intervals[i]
       ENDIF
    ENDLOOP
ENDDEFINE
```

```
DEFINE map_tasks_to_time_blocks(time_block_array,tasks_positions,mydayurl,todolisturl){
    DECLARE time_intervals[]
    time_intervals = ["6:00-7:00","7:00-8:00","8:00-9:00","9:00-10:00","10:00-11:00","11:00-12:00","12:00-13:00","13:00-14:00","14:00-15:00","15:00-16:00","16:00-17:00","17:00-18:00","18:00-19:00","19:00-20:00","20:00-21:00",
    "21:00-22:00","22:00-23:00","23:00-24:00","24:00-1:00","1:00-2:00","2:00-3:00","3:00-4:00","4:00-5:00","5:00-6:00"]

   DISPLAY "\nStep 3. Assign tasks to each work time block\n\n"
    DISPLAY "Time blocks:\n"
    DEFINE pos
    pos=0
    LOOP i=0;i<24;i++
       IF time_block_array[i]==1 THEN
           DISPLAY ++pos,time_intervals[i]
       ENDIF
    ENDLOOP

   DECLARE my_day_tasks[][]
    DECLARE my_day_tasks_last_index
    my_day_tasks_last_index=0

    DECLARE file
    file = OPEN FILE (todolisturl,"r");
    DECLARE task[]

    LOOP(i=0;END OF FILE;i++)
       READ FILE(file,task)
       LOOP(j=0;j<20;j++)
           IF (i == tasks_positions[j] && tasks_positions[j] != 0) THEN
               my_day_tasks[my_day_tasks_last_index] = task
               my_day_tasks_last_index++
               tasks_positions[j]=0
           ENDLOOP
       ENDLOOP
    ENDLOOP
    fclose(file);

    DISPLAY "Tasks LOOP today:\n"
    LOOP(i=0;i<20;i++){
       IF(my_day_tasks[i]!="") THEN
               DISPLAY i+1,my_day_tasks[i]
       ENDIF
    ENDLOOP

   DECLARE myday
   myday = OPEN FILE(mydayurl,"a")
    LOOP(i=0;i<24;i++)
       IF time_block_array[i]==1 THEN

           DECLARE tasks_to_incl_last_index,satisfactory_duration,tasks_to_include_indexes[]
           tasks_to_incl_last_index = 0
           satisfactory_duration=0;
           DOWHILE(satisfactory_duration==0)
```

(CONTINUED IN NEXT SCREENSHOT)

```
            DOWHILE(satisfactory_duration==0)

                DISPLAY "\nEnter tasks to allocate to time block %s (type 101 to when done)\n",time_intervals[i]

                DOWHILE(1)
                     READ INPUT tasks_to_include_indexes[tasks_to_incl_last_index++]

                     IF(tasks_to_include_indexes[tasks_to_incl_last_index-1]==101)
                        tasks_to_include_indexes[tasks_to_incl_last_index-1]=0
                        BREAK
                     ENDIF
                ENDDO


                DECLARE task_durations[20], task_durations_index,durations_sum,temp
                task_durations_index=0
                durations_sum=0

                LOOP(i=0;i<20;i++)
                    IF(tasks_to_include_indexes[i]==0) THEN
                        BREAK
                    ENDIF

                    READ FROM STRING(my_day_tasks[tasks_to_include_indexes[i]-1],temp[0],task_durations[task_durations_index++],temp[1],temp[2],temp[3] IN LOOPMAT "%s %d %s %s %s"
                ENDLOOP

                LOOP(i=0;i<20;i++)
                    durations_sum +=task_durations[i]
                ENDLOOP
                IF durations_sum>60 THEN
                    LOOP (i=0;i<20;i++)
                        tasks_to_include_indexes[i] = 0
                    ENDLOOP
                    tasks_to_incl_last_index = 0

                    DISPLAY "Your tasks LOOP this time block must be a maximum of 1 hour\n"
                ENDIF
                ELSE
                    satisfactory_duration=1;
                ENDELSE
            ENDDO

            LOOP(y=0;y<20;y++){
                IF(tasks_to_include_indexes[y]==0) THEN
                    CONTINUE
                }
             WRITE TO FILE(myday,time_intervals[i],"0%",my_day_tasks[tasks_to_include_indexes[y]-1]
            ENDLOOP
        ENDIF
      ENDLOOP
     CLOSE FILE(myday)
ENDEFINE
```

My day functions

```
DEFINE update_myday_task_progress(file_url[],line_pos, progress_value)
    DECLARE attribute_pos,row,file,temp
    attribute_pos = 1 char row[200]
     file = OPEN FILE(file_url,r+)
    temp = OPEN FILE("temp.txt",w+)

    LOOP(int i=0;END OF FILE(file);i++)
        READ INPUT(file,row)
        IF i==line_pos THEN
            DECLARE line_to_edit[][]
            READ FROM STRING(row,line_to_edit[0],line_to_edit[1],line_to_edit[2],line_to_edit[3],line_to_edit[4],line_to_edit[5],line_to_edit[6]) IN FORMAT"%s %s %s %s %s %s %s"

            DECLARE progress_value_string[4]
            itoa(progress_value,progress_value_string,10)
            progress_value_string = progress_value_string+"%"
            line_to_edit[attribute_pos] = progress_value_string

            WRITE TO FILE(temp,line_to_edit[0],line_to_edit[1],line_to_edit[2],line_to_edit[3],line_to_edit[4],line_to_edit[5],line_to_edit[6])

       ENDIF
        ELSE
            IF (END OF FILE(file)) THEN
                BREAK
            ENDIF
            WRITE TO FILE(temp,row)
        ENDELSE
    ENDLOOP

    CLOSE FILE(file)
    CLOSE FILE(temp)

    REMOVE(file_url)
    RENAME("temp.txt",file_url)
ENDDEFINE
```

```
DEFINE delete_myday_task(file_url[],line_pos) THEN
    DECLARE row[],file,temp
    file = OPEN FILE(file_url,r)
    temp = OPEN FILE("temp.txt",w+)

    LOOP(i=0;!END OF FILE(file);i++)
        READ FILE LINE(fil,row)
        IF(i==line_pos) THEN
           CONTINUE
        ENDIF
        ELSE{
            IF (END OF FILE(file)) THEN
                BREAK
            ENDIF
            fprintf(temp,"%s",row);
            WRITE TO FILE(temp,row)
        ENDELSE
    ENDLOOP
    CLOSE FILE(file)
    CLOSE FILE(temp)

    REMOVE(file_url)
    RENAME("temp.txt",file_url)
    RETURN 0
ENDDEFINE
```

```
DEFINE my_day(myday_file_url[])
    DISPLAY "\nMy day tasks\n"
    DECLARE num_tasks
    num_tasks = display_tasks(myday_file_url)

    DISPLAY "Which task do you want to select\n"
    DECLARE task_pos
    READ INPUT task_pos

    IF(task_pos<=0) THEN
        DISPLAY "Error: Invalid task position\n"
        RETURN -1
    ENDIF
    ELSE IF(task_pos>num_tasks) THEN
        DISPLAY "Error:Task position does not exist\n"
        RETURN -1
    ENDIF

    DISPLAY "Enter progress of task\n"
    DECLARE progress
    READ INPUT(progress)

    IF progress>100 ||  progress<0 THEN
        DISPLAY "Progress must be a value between 0 and 100\n"
        RETURN -1
    ENDIF

    IF(progress!=100) THEN
        update_myday_task_progress(myday_file_url,task_pos,progress);
    ENDIF
    ELSE
        delete_myday_task(myday_file_url,task_pos);

    ENDELSE
    DISPLAY "Successful\n"
    RETURN 0
ENDEFINE
```

Miscellaneous functions

```
DEFINE compare_dates(date1[],date2[]) THEN
    DECLARE yr1,yr2,mnth1,mnth2,day1,day2
    READ FROM STRING (date1,day1,mnth1,yr1)
    READ FROM STRING(date2,&day2,&mnth2,&yr2)

    IF(yr1>yr2) THEN
        RETURN 1
    ENDIF
    ELSEIF(yr1<yr2) THEN
        RETURN -1
    ENDIF
    ELSEIF(yr1 == yr2) THEN
        IF(mnth1>mnth2) THEN
            RETURN 1
        ENDIF
        ELSEIF(mnth1<mnth2) THEN
            RETURN -1
        ENDIF
        ELSEIF(mnth1 == mnth2) THEN
            IF(day1>day2) THEN
                RETURN 1
            ELSEIF(day1<day2) THEN
                RETURN -1
            ENDIF
            ELSEIF(day1 == day2) THEN
                RETURN 0
            ENDIF
        ENDIF
    ENDIF
ENDDEFINE
```

```
DEFINE generate_date()
    DECLARE time_string,time,tm
    t = time(NULL)
    tm = localtime(t)
    DISPLAY FROM STRING (time_string,tm.tm_mday,tm.tm_mon + 1,tm.tm_year + 1900) IN FORMAT "%02d/%02d/%d"
    RETURN time_string
ENDDEFINE
```

```
DEFINE display_tasks(file_url[])
    DECLARE file
    file=OPEN FILE(file_url,r)
    DECLARE contents[]
    DECLARE num_lines
    num_lines=0

    LOOP (c=0;READ FILE LINE(file,contents);c++)
        IF c!=0 THEN
            DISPLAY(c,contents)
            DISPLAY("\n")
        ENDIF
        num_lines++
    ENDLOOP
    CLOSE FILE(file)
    RETURN num_lines
ENDDEFINE
```

```
DEFINE validate_duration(value[])
    IF(value!=0 OR value!=15 OR value!=30 OR value!=45; value!=60)
        RETURN -1
    ENDIF
    RETURN 0
ENDDEFINE
```

```
DEFINE selectionSort(char arr[50][5][30], int n,int attr_pos,int asc){
    DECLARE i, j, min_idx
    LOOP (i = 0; i < n - 1; i++)
        min_idx = i
        LOOP (j = i + 1; j < n; j++)
            IF(asc==1) THEN
                IF (arr[j][attr_pos]<arr[min_idx][attr_pos]) THEN
                    min_idx = j
                ENDIF
            ENDIF
            ELSEIF(asc==0) THEN
                IF (arr[j][attr_pos]>arr[min_idx][attr_pos]) THEN
                    min_idx = j
                ENDIF
            ENDIF
        ENDLOOP

        DECLARE temp[][]
        LOOP(k=0;k<5;k++)
            temp[k]=arr[i][k]
        ENDLOOP
        LOOP(k=0;k<5;k++){
            arr[i][k]=arr[min_idx][k]
        ENDLOOP
        LOOP(k=0;k<5;k++){
            arr[min_idx][k]=temp[k]
        ENDLOOP
    ENDLOOP

    LOOP(a=0;a<50;a++){
        IF arr[a]=="" THEN
            CONTINUE
        ENDIF
        LOOP(b=0;b<5;b++)
            DISPLAY "%s ",arr[a][b]
        ENDLOOP
        DISPLAY "\n"
    ENDLOOP
ENDEFINE
```

```
DEFINE selectionSortDate(char arr[50][5][30], int n,int attr_pos,int asc){
    DECLARE i, j, min_idx
    LOOP (i = 0; i < n - 1; i++)
        min_idx = i
        LOOP (j = i + 1; j < n; j++)
            IF(asc==1) THEN
                IF (compare_dates(arr[j][attr_pos],arr[min_idx][attr_pos])<0) THEN
                    min_idx = j
                ENDIF
            ENDIF
            ELSEIF(asc==0) THEN
                IF (compare_dates(arr[j][attr_pos],arr[min_idx][attr_pos])>0) THEN
                    min_idx = j
                ENDIF
            ENDIF
        ENDLOOP

        DECLARE temp[][]
        LOOP(k=0;k<5;k++)
            temp[k]=arr[i][k]
        ENDLOOP
        LOOP(k=0;k<5;k++){
            arr[i][k]=arr[min_idx][k]
        ENDLOOP
        LOOP(k=0;k<5;k++){
            arr[min_idx][k]=temp[k]
        ENDLOOP
    ENDLOOP

    LOOP(a=0;a<50;a++){
        IF arr[a]=="" THEN
            CONTINUE
        ENDIF
        LOOP(b=0;b<5;b++)
            DISPLAY "%s ",arr[a][b]
        ENDLOOP
        DISPLAY "\n"
    ENDLOOP
ENDEFINE
```

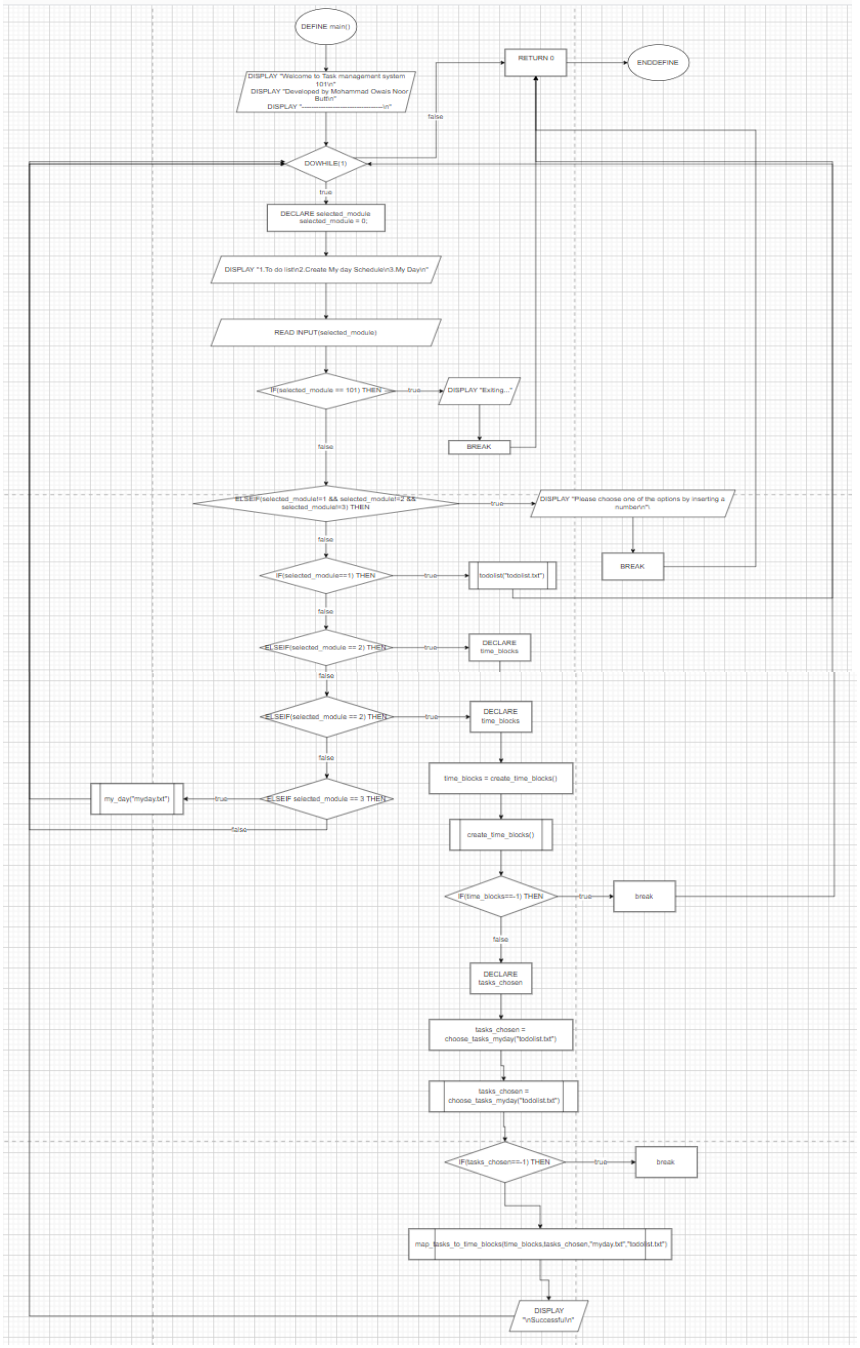
```
DEFINE validate_date(char time_string[8])
    DECLARE day,month,year
    READ FROM STRING(time_string,&day,&month,&year) IN FORMAT "%d/%d/%d"

    IF(year<=2022 && year>=2122) THEN
        DISPLAY "Invalid year\n"
        RETURN -1
    ENDIF
    IF(month == 1 || month == 3 || month == 5 || month == 7 || month == 8 || month == 10 || month == 12) THEN
        IF(!(day>0 && day<=31)) THEN
            DISPLAY("invalid day\n");
            RETURN -1
        ENDIF
    ENDIF
    ELSEIF(month == 4 || month == 6 || month == 9 || month == 11) THEN
        IF(!(day>0 && day<=30)) THEN
            DISPLAY("invalid day\n")
            RETURN -1
        ENDIF
    ENDIF
        IF(year%4==0) THEN
                IF !(day>0 && day<=28) THEN
                    DISPLAY "invalid day\n"
                    RETURN -1
                ENDIF
        ENDIF
        ELSE{
            IF(!(day>0 && day<=29)) THEN
                DISPLAY("invalid day\n")
                RETURN -1
                ENDIF
        ENDELSE

    ENDIF
    ELSE
        DISPLAY("invalid month\n");
        RETURN -1
    ENDELSE

    RETURN 0
ENDDEFINE
```

Main

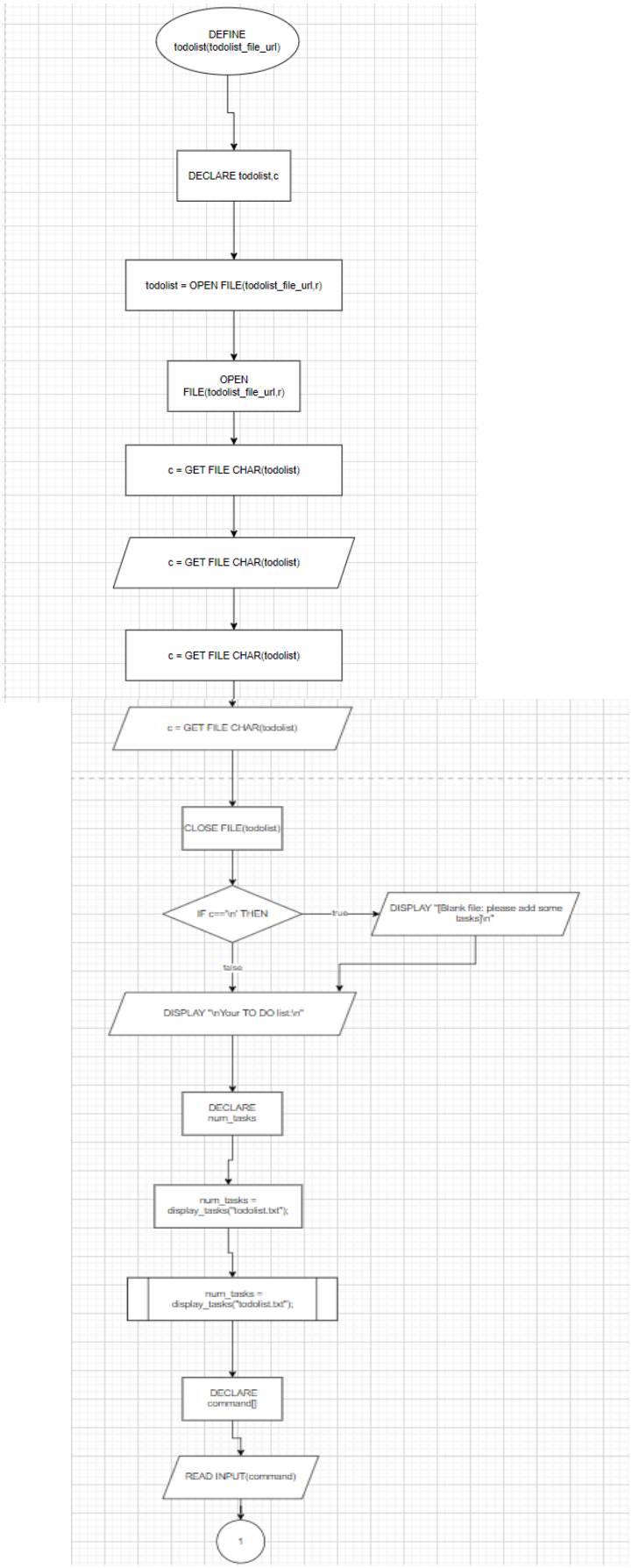
```
BEGIN
    DISPLAY "Welcome to Task management system 101\n"
    DISPLAY "Developed by Mohammad Owais Noor Butt\n"
    DISPLAY "----------------------------------\n"
    DOWHILE(1){
        DECLARE selected_module
        selected_module = 0;

        DISPLAY "1.To do list\n2.Create My day Schedule\n3.My Day\n"
        READ INPUT(selected_module)


        IF(selected_module == 101) THEN
            DISPLAY "Exiting..."
            BREAK
        ENDIF
        ELSEIF(selected_module!=1 && selected_module!=2 && selected_module!=3) THEN
            DISPLAY "Please choose one of the options by inserting a number\n"\
            BREAK
        ENDIF

        IF(selected_module==1) THEN
            todolist("todolist.txt")
        ENDIF
        ELSEIF(selected_module == 2) THEN
            DECLARE time_blocks
            time_blocks = create_time_blocks()
            IF(time_blocks==-1) THEN
                BREAK
            ENDIF

            DECLARE tasks_chosen
            tasks_chosen = choose_tasks_myday("todolist.txt")
            IF(tasks_chosen==-1) THEN
                BREAK
            ENDIF

            map_tasks_to_time_blocks(time_blocks,tasks_chosen,"myday.txt","todolist.txt")
            DISPLAY "\nSuccessful\n"


        ENDIF
        ELSEIF selected_module == 3 THEN
            my_day("myday.txt")
        ENDIF
    ENDWHILE
    RETURN 0
END
```

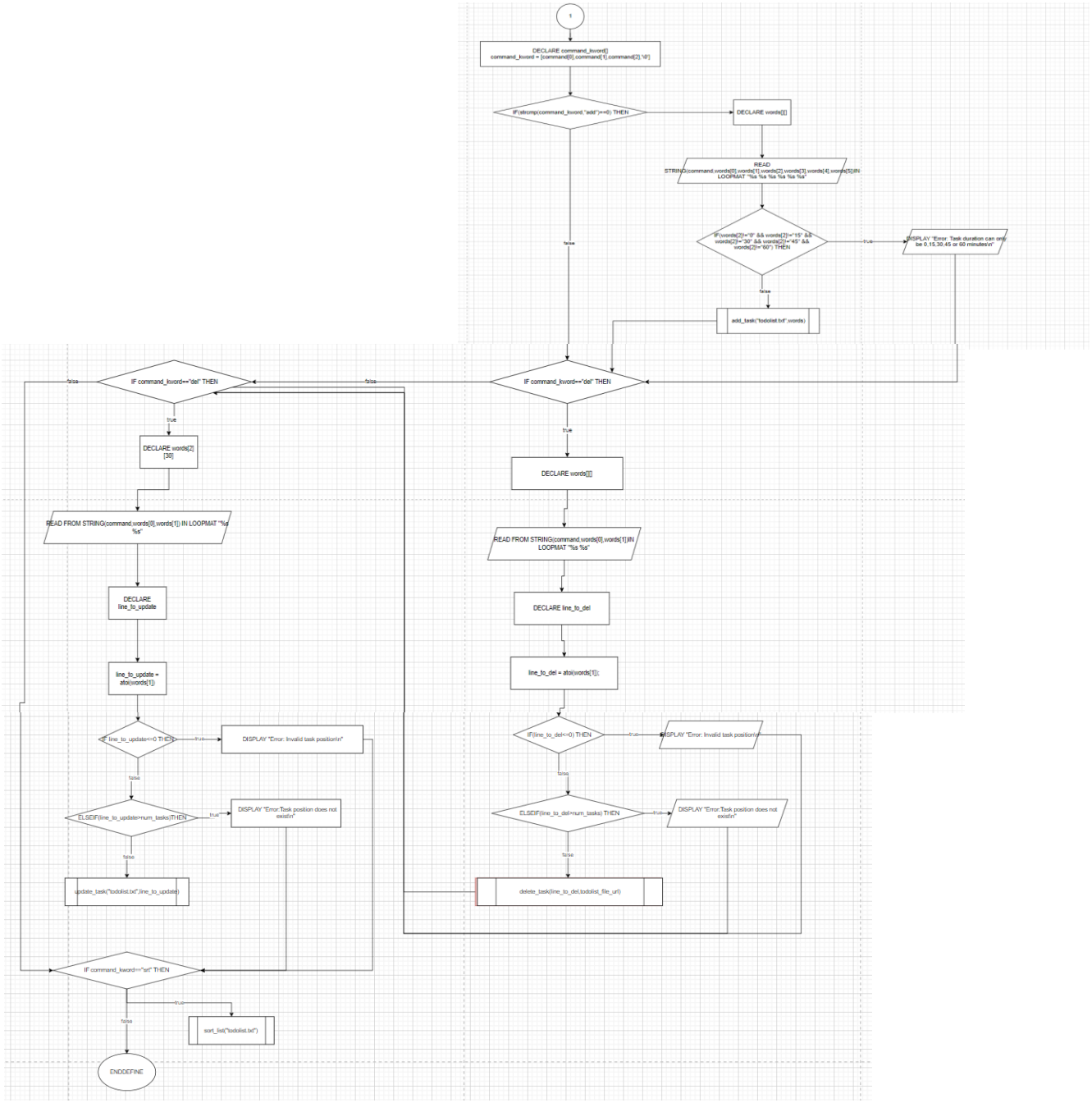
Design of the code – Flowcharts

Main flow

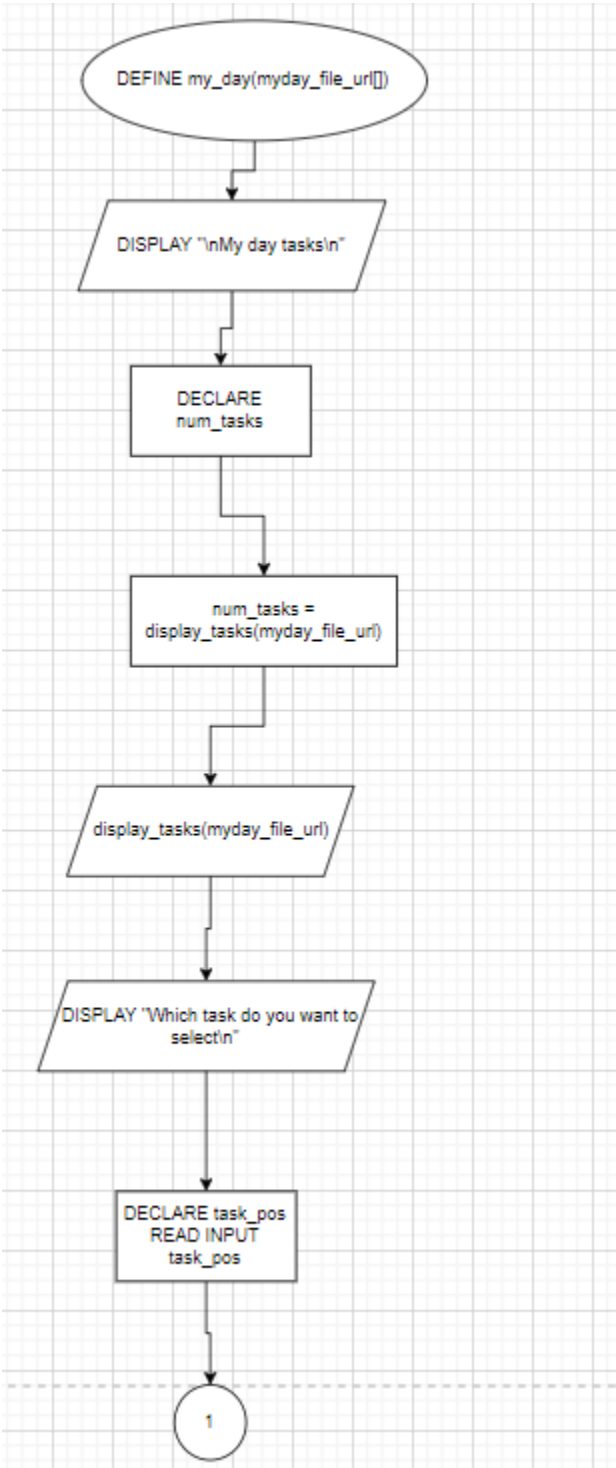


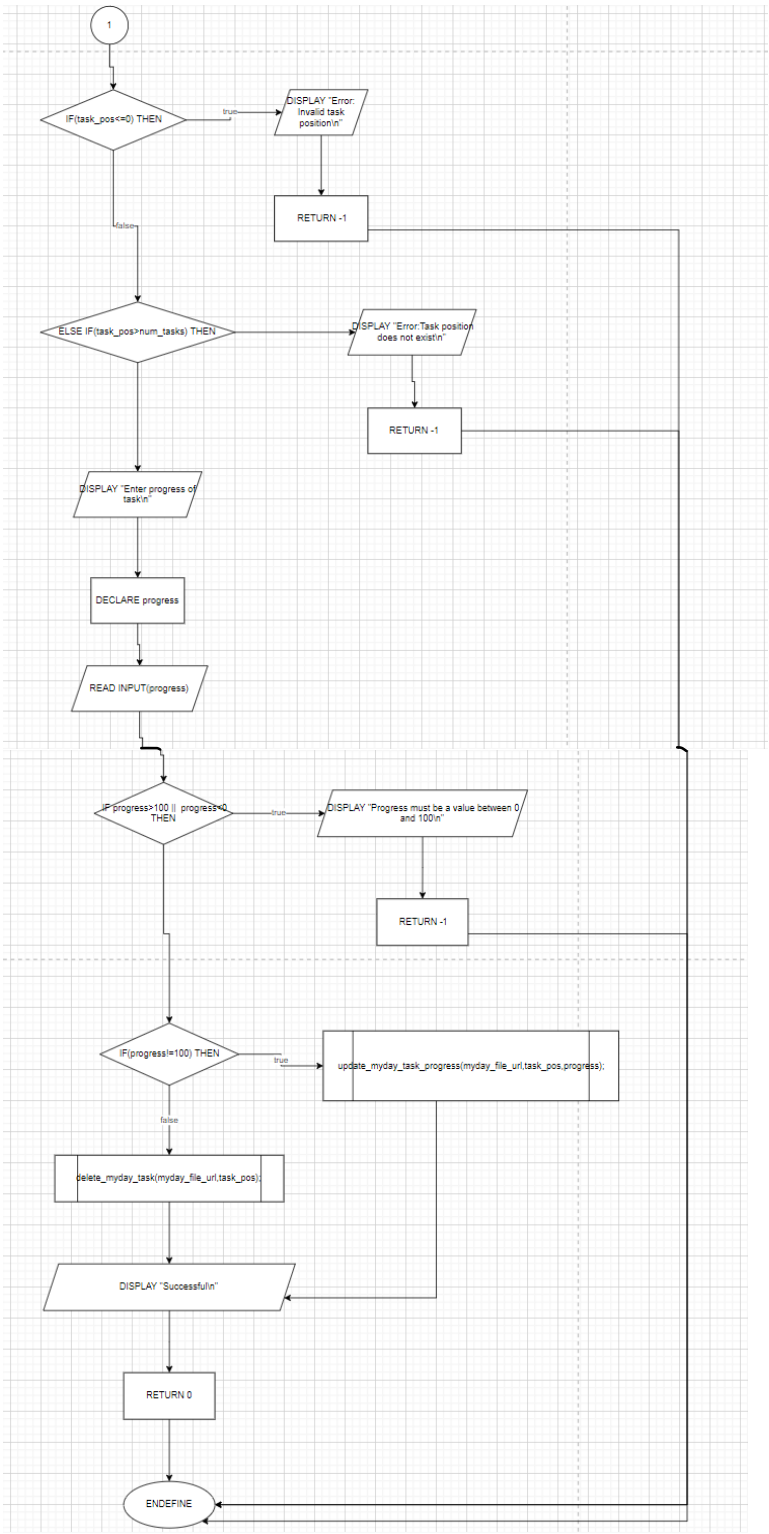
Todolist function


DEFINE todolist(todolist_file_url)
DECLARE todolist,c
todolist = OPEN FILE(todolist_file_url,r)
OPEN FILE(todolist_file_url,r)
c = GET FILE CHAR(todolist)
c = GET FILE CHAR(todolist)
c = GET FILE CHAR(todolist)
c = GET FILE CHAR(todolist)
CLOSE FILE(todolist)
IF c=='\n' THEN
true
DISPLAY "[Blank file: please add some tasks]\n"
false
DISPLAY "\nYour TO DO list:\n"
DECLARE num_tasks
num_tasks = display_tasks("todolist.txt");
num_tasks = display_tasks("todolist.txt");
DECLARE command[]
READ INPUT(command)
1


1
DECLARE command_kword[]
command_kword = [command[0],command[1],command[2],'\0']
IF(strcmp(command_kword,"add")==0) THEN
DECLARE words[][]
READ STRING(command,words[0],words[1],words[2],words[3],words[4],words[5])IN LOOPMAT "%s %s %s %s %s %s"
IF(words[2]!='0' && words[2]!='15' && words[2]!='30' && words[2]!='45' && words[2]!='60') THEN
DISPLAY "Error: Task duration can only be 0,15,30,45 or 60 minutes\n"
add_task("todolist.txt",words)
IF command_kword=="del" THEN
DECLARE words[][]
READ FROM STRING(command,words[0],words[1])IN LOOPMAT "%s %s"
DECLARE line_to_del
line_to_del = atoi(words[1]);
IF(line_to_del<=0) THEN
DISPLAY "Error: Invalid task position\n"
ELSEIF(line_to_del>num_tasks) THEN
DISPLAY "Error:Task position does not exist\n"
delete_task(line_to_del,todolist_file_url)
IF command_kword=="del" THEN
DECLARE words[2][30]
READ FROM STRING(command,words[0],words[1]) IN LOOPMAT "%s %s"
DECLARE line_to_update
line_to_update = atoi(words[1])
IF line_to_update<=0 THEN
DISPLAY "Error: Invalid task position\n"
ELSEIF(line_to_update>num_tasks)THEN
DISPLAY "Error:Task position does not exist\n"
update_task("todolist.txt",line_to_update)
IF command_kword=="srt" THEN
sort_list("todolist.txt")
ENDDEFINE
true
false

My day function




1
IF(task_pos<=0) THEN
true
DISPLAY "Error: Invalid task position\n"
RETURN -1
false
ELSE IF(task_pos>num_tasks) THEN
DISPLAY "Error:Task position does not exist\n"
RETURN -1
DISPLAY "Enter progress of task\n"
DECLARE progress
READ INPUT(progress)
IF progress>100 || progress<0 THEN
true
DISPLAY "Progress must be a value between 0 and 100\n"
RETURN -1
IF(progress!=100) THEN
true
update_myday_task_progress(myday_file_url,task_pos,progress);
false
delete_myday_task(myday_file_url,task_pos);
DISPLAY "Successful\n"
RETURN 0
ENDEFINE

Additional features

```c
char * generate_date(){
    time_t t = time(NULL);
    struct tm tm = *localtime(&t);
    static char time_string[10];
    sprintf(time_string,"%02d/%02d/%d", tm.tm_mday,tm.tm_mon + 1,tm.tm_year + 1900);

    return time_string;
    }
```

The generate_date function uses the time.h library. It first declares time_t type variable which stores the number of seconds since epoch time. This variable is assigned to the current time in the system.

The t variable is converted into the local time zone using localtime() and assigned to the tm variable.

This variable is the converted to a string.

Screenshots of code

Main menu

```
Welcome to Task management system 101
Developed by Mohammad Owais Noor Butt
---------------------------------
1.To do list
2.Create My day Schedule
3.My Day
```

Figure 1

The user is greeted with a welcome screen and a main menu is displayed. The user can select the required option by typing in a number.

To do list option

This option allows the user to view their to do list, add tasks to it, delete tasks from it, update task values and sort tasks.

```
1.To do list
2.Create My day Schedule
3.My Day
1

Your TO DO list:
1. icp_assignment 60 22/06/2022 ICP 22/06/2022

2. idb_db_schema 30 22/06/2022 IDB 22/06/2022

3. BMK_video_presentation 45 30/06/2022 IDB 22/06/2022
```

Figure 2

If the user types 1, their TO DO list is displayed where they can type commands at the screen to make any modifications or adjustments.

```
Your TO DO list:
1. icp_assignment 60 22/06/2022 ICP 22/06/2022

2. idb_db_schema 30 22/06/2022 IDB 22/06/2022

3. BMK_video_presentation 45 30/06/2022 IDB 22/06/2022

add feed_goldfish 15 23/06/2022 Normal
1.To do list
2.Create My day Schedule
3.My Day
1

Your TO DO list:
1. icp_assignment 60 22/06/2022 ICP 22/06/2022

2. idb_db_schema 30 22/06/2022 IDB 22/06/2022

3. BMK_video_presentation 45 30/06/2022 IDB 22/06/2022

4. feed_goldfish 15 23/06/2022 Normal 22/06/2022
```

Figure 3

The add command is used to add tasks to the to do list. As can be seen, the description, duration, due date, category and date created of the tasks have been input. The new task is added after the execution of the command with an automatically generated “date created” attribute at the end

```
Your TO DO list:
1. icp_assignment 60 22/06/2022 ICP 22/06/2022

2. idb_db_schema 30 22/06/2022 IDB 22/06/2022

3. BMK_video_presentation 45 30/06/2022 IDB 22/06/2022

4. feed_goldfish 15 23/06/2022 Normal 22/06/2022

del 4
1.To do list
2.Create My day Schedule
3.My Day
1

Your TO DO list:
1. icp_assignment 60 22/06/2022 ICP 22/06/2022

2. idb_db_schema 30 22/06/2022 IDB 22/06/2022

3. BMK_video_presentation 45 30/06/2022 IDB 22/06/2022
```

Figure 4

Tasks can also be deleted using the del command, the line number of the task is specified.

```
Your TO DO list:
1. icp_assignment 60 22/06/2022 ICP 22/06/2022

2. idb_db_schema 30 22/06/2022 IDB 22/06/2022

3. BMK_video_presentation 45 30/06/2022 IDB 22/06/2022

upd 3
Which attribute do you want to update?
1.Task description
2.Duration
3.Due date
4.Category
2
What is the new value?
30
1.To do list
2.Create My day Schedule
3.My Day
1

Your TO DO list:
1. icp_assignment 60 22/06/2022 ICP 22/06/2022

2. idb_db_schema 30 22/06/2022 IDB 22/06/2022

3. BMK_video_presentation 30 30/06/2022 IDB 22/06/2022
```

Figure 5

Tasks can also be updated in case a mistake has been made during entry or the task has changed, the upd command is input followed by the position of the task. Next the user is asked which attribute they would like to select. In this case the duration is chose and is updated to 30 minutes. This change evident in the newly generated to do list.

```
Your TO DO list:
1. icp_assignment 60 22/06/2022 ICP 22/06/2022

2. idb_db_schema 30 22/06/2022 IDB 22/06/2022

3. BMK_video_presentation 30 30/06/2022 IDB 22/06/2022

srt
Sort by:
1.Task description
2.Duration
3.Due date
4.Category
5.Date created
1
What order? Ascending=1;Descending=0;
1

BMK_video_presentation 30 30/06/2022 IDB 22/06/2022
BMK_video_presentation 30 30/06/2022 IDB 22/06/2022
icp_assignment 60 22/06/2022 ICP 22/06/2022
idb_db_schema 30 22/06/2022 IDB 22/06/2022

1.To do list
2.Create My day Schedule
3.My Day
```

Figure 6

The srt command is used to sort the tasks for reference. After entering the command, the user is asked which attribute they want to sort by and in which order. The srt command does not modify the original to do list.

```
Your TO DO list:
1. icp_assignment 60 22/06/2022 ICP 22/06/2022

2. idb_db_schema 30 22/06/2022 IDB 22/06/2022

3. BMK_video_presentation 30 30/06/2022 IDB 22/06/2022

srt
Sort by:
1.Task description
2.Duration
3.Due date
4.Category
5.Date created
3
What order? Ascending=1;Descending=0;
0

BMK_video_presentation 30 30/06/2022 IDB 22/06/2022
BMK_video_presentation 30 30/06/2022 IDB 22/06/2022
icp_assignment 60 22/06/2022 ICP 22/06/2022
idb_db_schema 30 22/06/2022 IDB 22/06/2022

1.To do list
2.Create My day Schedule
3.My Day
```

Figure 7

In this screenshot, the srt command is used to order the to do list in descending order of due dates

Create my day Schedule

This option allows the user to time block their tasks. Time blocking is a technique where tasks are assigned timings from 24 hours in a day, it is one of the best time management techniques available. The create my day schedule allows the user to choose tasks for the day from their **to do list** and adds it to the **my day** list together with the timings.

The process has 3 steps:

- Choose work time blocks from a 24-hour day
- Choose the tasks to include from to do list
- Allocated tasks chosen to work time blocks

```
1.To do list
2.Create My day Schedule
3.My Day
2
Step 1. Time blocks in a day
1 6:00 - 7:00 Rest
2 7:00 - 8:00 Rest
3 8:00 - 9:00 Rest
4 9:00 - 10:00 Rest
5 10:00 - 11:00 Rest
6 11:00 - 12:00 Rest
7 12:00 - 13:00 Rest
8 13:00 - 14:00 Rest
9 14:00 - 15:00 Rest
10 15:00 - 16:00 Rest
11 16:00 - 17:00 Rest
12 17:00 - 18:00 Rest
13 18:00 - 19:00 Rest
14 19:00 - 20:00 Rest
15 20:00 - 21:00 Rest
16 21:00 - 22:00 Rest
17 22:00 - 23:00 Rest
18 23:00 - 24:00 Rest
19 24:00 - 1:00 Rest
20 1:00 - 2:00 Rest
21 2:00 - 3:00 Rest
22 3:00 - 4:00 Rest
23 4:00 - 5:00 Rest
24 5:00 - 6:00 Rest
Select time block (press 101 when you are done):
```

Figure 8

Figure 9

When the second option is chosen from the main menu, the time blocks in a day are generated and displayed. Each time block is 1 hour.

```
Select time block (press 101 when you are done):
2
Is the time rest or work? R/W
w
Step 1. Time blocks in a day
1 6:00 - 7:00 Rest
2 7:00 - 8:00 Work
3 8:00 - 9:00 Rest
4 9:00 - 10:00 Rest
5 10:00 - 11:00 Rest
6 11:00 - 12:00 Rest
7 12:00 - 13:00 Rest
8 13:00 - 14:00 Rest
9 14:00 - 15:00 Rest
10 15:00 - 16:00 Rest
11 16:00 - 17:00 Rest
12 17:00 - 18:00 Rest
13 18:00 - 19:00 Rest
14 19:00 - 20:00 Rest
15 20:00 - 21:00 Rest
16 21:00 - 22:00 Rest
17 22:00 - 23:00 Rest
18 23:00 - 24:00 Rest
19 24:00 - 1:00 Rest
20 1:00 - 2:00 Rest
21 2:00 - 3:00 Rest
22 3:00 - 4:00 Rest
23 4:00 - 5:00 Rest
24 5:00 - 6:00 Rest
Select time block (press 101 when you are done):
```

Figure 10

Figure 11

The user can choose whether the time block is a work or rest time block by entering the number of the time block and choosing the appropriate category. As can be seen above, time block 2 has been changed to a work time block.

```
Step 1. Time blocks in a day
1 6:00 - 7:00 Rest
2 7:00 - 8:00 Work
3 8:00 - 9:00 Work
4 9:00 - 10:00 Work
5 10:00 - 11:00 Rest
6 11:00 - 12:00 Rest
7 12:00 - 13:00 Rest
8 13:00 - 14:00 Rest
9 14:00 - 15:00 Rest
10 15:00 - 16:00 Rest
11 16:00 - 17:00 Rest
12 17:00 - 18:00 Rest
13 18:00 - 19:00 Rest
14 19:00 - 20:00 Rest
15 20:00 - 21:00 Rest
16 21:00 - 22:00 Rest
17 22:00 - 23:00 Rest
18 23:00 - 24:00 Rest
19 24:00 - 1:00 Rest
20 1:00 - 2:00 Rest
21 2:00 - 3:00 Rest
22 3:00 - 4:00 Rest
23 4:00 - 5:00 Rest
24 5:00 - 6:00 Rest
Select time block (press 101 when you are done):
```

Figure 12

Figure 13

More work time blocks have been added (time blocks 2,3 and 4). Say for example the user does not want time block 4 to be a work time block, they can be reverse it as shown by the following screenshot

```
Select time block (press 101 when you are done):
4
Is the time rest or work? R/W
r
Step 1. Time blocks in a day
1 6:00 - 7:00 Rest
2 7:00 - 8:00 Work
3 8:00 - 9:00 Work
4 9:00 - 10:00 Rest
5 10:00 - 11:00 Rest
6 11:00 - 12:00 Rest
7 12:00 - 13:00 Rest
8 13:00 - 14:00 Rest
9 14:00 - 15:00 Rest
10 15:00 - 16:00 Rest
11 16:00 - 17:00 Rest
12 17:00 - 18:00 Rest
13 18:00 - 19:00 Rest
14 19:00 - 20:00 Rest
15 20:00 - 21:00 Rest
16 21:00 - 22:00 Rest
17 22:00 - 23:00 Rest
18 23:00 - 24:00 Rest
19 24:00 - 1:00 Rest
20 1:00 - 2:00 Rest
21 2:00 - 3:00 Rest
22 3:00 - 4:00 Rest
23 4:00 - 5:00 Rest
24 5:00 - 6:00 Rest
Select time block (press 101 when you are done):
```

Figure 14

They chose time block for and type in the letter ‘R’ for rest.

```
Select time block (press 101 when you are done):
101
1 6:00 - 7:00 Rest
2 7:00 - 8:00 Work
3 8:00 - 9:00 Work
4 9:00 - 10:00 Rest
5 10:00 - 11:00 Rest
6 11:00 - 12:00 Rest
7 12:00 - 13:00 Rest
8 13:00 - 14:00 Rest
9 14:00 - 15:00 Rest
10 15:00 - 16:00 Rest
11 16:00 - 17:00 Rest
12 17:00 - 18:00 Rest
13 18:00 - 19:00 Rest
14 19:00 - 20:00 Rest
15 20:00 - 21:00 Rest
16 21:00 - 22:00 Rest
17 22:00 - 23:00 Rest
18 23:00 - 24:00 Rest
19 24:00 - 1:00 Rest
20 1:00 - 2:00 Rest
21 2:00 - 3:00 Rest
22 3:00 - 4:00 Rest
23 4:00 - 5:00 Rest
24 5:00 - 6:00 Rest
Your workhours for the day: 2 hours

TO DO list
1. icp_assignment 60 22/06/2022 ICP 22/06/2022

2. idb_db_schema 30 22/06/2022 IDB 22/06/2022

3. BMK_video_presentation 30 30/06/2022 IDB 22/06/2022

Step2. Choose task(101 to exit)
```

Figure 15

The next step is to choose the tasks from the to do list that the user wants to perform in their day. The to do list is displayed. The workhours for the day are also displayed

```
TO DO list
1. icp_assignment 60 22/06/2022 ICP 22/06/2022

2. idb_db_schema 30 22/06/2022 IDB 22/06/2022

3. BMK_video_presentation 30 30/06/2022 IDB 22/06/2022

Step2. Choose task(101 to exit)
1
2
3
101

Step 3. Assign tasks to each work time block

Time blocks:
1. 7:00-8:00
2. 8:00-9:00
Tasks for today:
1. icp_assignment 60 22/06/2022 ICP 22/06/2022
2. idb_db_schema 30 22/06/2022 IDB 22/06/2022
3. BMK_video_presentation 30 30/06/2022 IDB 22/06/2022

Enter tasks to allocate to time block 7:00-8:00 (type 101 to when done)
```

Figure 16

The user chooses tasks from the to do list by entering tasks positions and entering 101 to exit. The work time blocks and tasks chosen appear after that, and the user is prompted to choose tasks for the first time block.

```
Enter tasks to allocate to time block 7:00-8:00 (type 101 to when done)
1
101

Enter tasks to allocate to time block 8:00-9:00 (type 101 to when done)
2
3
101

Successful
```

Figure 17

The tasks are entered and 101 is input at the end to tell the program to go to the next time block.

Note, that two or more tasks can be entered for one time block as shown in the second user input.

My day

The my day option allows the user to view their tasks for the day and update the progress.

```
1.To do list
2.Create My day Schedule
3.My Day
3

My day tasks
1. 7:00-8:00 0% icp_assignment 60 22/06/2022 ICP 22/06/2022

2. 8:00-9:00 0% idb_db_schema 30 22/06/2022 IDB 22/06/2022

3. 8:00-9:00 0% BMK_video_presentation 30 30/06/2022 IDB 22/06/2022

Which task do you want to select
```

Figure 18

The tasks that were chosen in the ‘create my day schedule” option, As can be seen the default progress is 0%.

```
Which task do you want to select
1
Enter progress of task
50
Successful
1.To do list
2.Create My day Schedule
3.My Day
3

My day tasks
1. 7:00-8:00 50% icp_assignment 60 22/06/2022 ICP 22/06/2022

2. 8:00-9:00 0% idb_db_schema 30 22/06/2022 IDB 22/06/2022

3. 8:00-9:00 0% BMK_video_presentation 30 30/06/2022 IDB 22/06/2022

Which task do you want to select
```

Figure 19

Tasks can be chosen by inputting the task position. Next the percentage is entered, in this case 50.

```
Which task do you want to select
3
Enter progress of task
100
Successful
1.To do list
2.Create My day Schedule
3.My Day
3

My day tasks
1. 7:00-8:00 50% icp_assignment 60 22/06/2022 ICP 22/06/2022

2. 8:00-9:00 0% idb_db_schema 30 22/06/2022 IDB 22/06/2022

Which task do you want to select
```

Figure 20

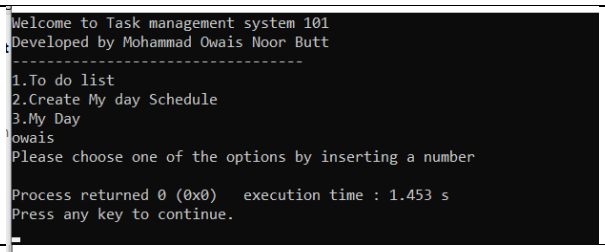
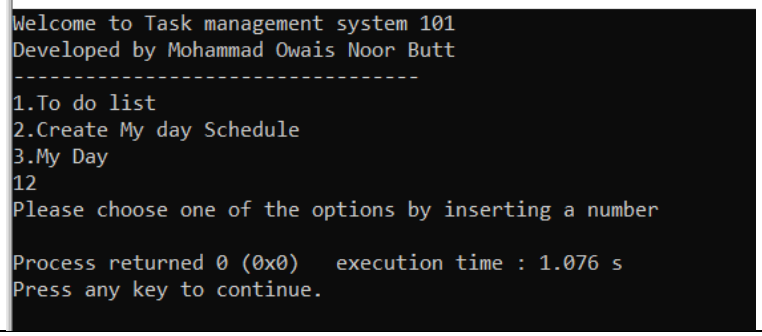
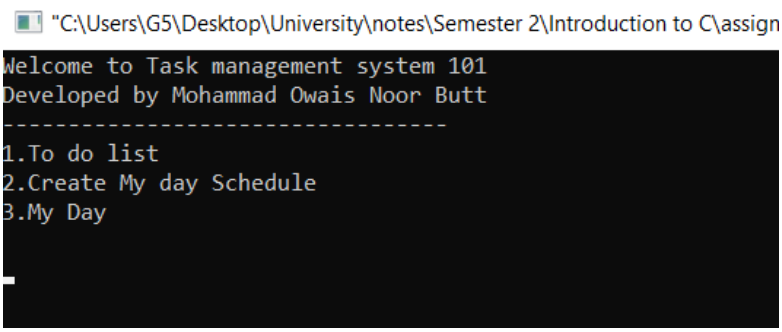
If the percentage entered is 100%, the task is removed from the my day tasks list.

Test plan

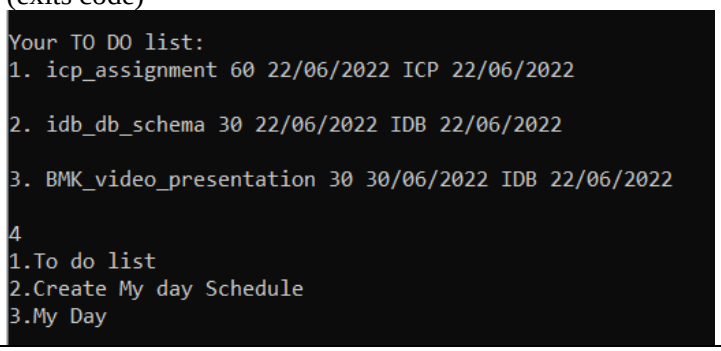
The test plan will test the system based on the screens provided in the "screenshots of code" section. Several types of Input will be provided and the output will be recorded.

Main menu

Input	Output	Pass/Fail
1	1.To do list 2.Create My day Schedule 3.My Day 1 Your TO DO list: 1. icp_assignment 60 22/06/2022 ICP 22/06/2022 2. idb_db_schema 30 22/06/2022 IDB 22/06/2022 3. BMK_video_presentation 45 30/06/2022 IDB 22/06/2022	pass
0	1.To do list 2.Create My day Schedule 3.My Day 0 Please choose one of the options by inserting a number Process returned 0 (0x0) execution time : 10.960 s Press any key to continue.	pass

text		Pass
12		pass
space	 (does nothing)	pass

To do list option

Input	Output	Pass/Fail
Out of limit value	(exits code) 	pass

text	```Your TO DO list: 1. icp_assignment 60 22/06/2022 ICP 22/06/2022 2. idb_db_schema 30 22/06/2022 IDB 22/06/2022 3. BMK_video_presentation 30 30/06/2022 IDB 22/06/2022 owais 1.To do list 2.Create My day Schedule 3.My Day``` (exits)	pass
Add command	```Your TO DO list: 1. icp_assignment 60 22/06/2022 ICP 22/06/2022 2. idb_db_schema 30 22/06/2022 IDB 22/06/2022 3. BMK_video_presentation 45 30/06/2022 IDB 22/06/2022 add feed_goldfish 15 23/06/2022 Normal 1.To do list 2.Create My day Schedule 3.My Day 1 Your TO DO list: 1. icp_assignment 60 22/06/2022 ICP 22/06/2022 2. idb_db_schema 30 22/06/2022 IDB 22/06/2022 3. BMK_video_presentation 45 30/06/2022 IDB 22/06/2022 4. feed_goldfish 15 23/06/2022 Normal 22/06/2022``` 	

Invalid due date for add command	```Your TO DO list:
	1. icp_assignment 60 22/06/2022 ICP 22/06/2022
	
	2. idb_db_schema 30 22/06/2022 IDB 22/06/2022
	
	3. BMK_video_presentation 30 30/06/2022 IDB 22/06/2022
	
	add task1 30 33/02/2022 RRR
	invalid day
	1.To do list
	2.Create My day Schedule
	3.My Day
	``
	
		pass

Del statement	Your TO DO list: 1. icp_assignment 60 22/06/2022 ICP 22/06/2022 2. idb_db_schema 30 22/06/2022 IDB 22/06/2022 3. BMK_video_presentation 45 30/06/2022 IDB 22/06/2022 4. feed_goldfish 15 23/06/2022 Normal 22/06/2022 del 4 1.To do list 2.Create My day Schedule 3.My Day 1 Your TO DO list: 1. icp_assignment 60 22/06/2022 ICP 22/06/2022 2. idb_db_schema 30 22/06/2022 IDB 22/06/2022 3. BMK_video_presentation 45 30/06/2022 IDB 22/06/2022	
Invalid input for del command	Your TO DO list: 1. icp_assignment 60 22/06/2022 ICP 22/06/2022 2. idb_db_schema 30 22/06/2022 IDB 22/06/2022 3. BMK_video_presentation 30 30/06/2022 IDB 22/06/2022 del 9 Error:Task position does not exist	pass
Text used for del command input	Your TO DO list: 1. icp_assignment 60 22/06/2022 ICP 22/06/2022 2. idb_db_schema 30 22/06/2022 IDB 22/06/2022 3. BMK_video_presentation 30 30/06/2022 IDB 22/06/2022 del owais	pass
Input for upd command	Your TO DO list: 1. icp_assignment 60 22/06/2022 ICP 22/06/2022 2. idb_db_schema 30 22/06/2022 IDB 22/06/2022 3. BMK_video_presentation 45 30/06/2022 IDB 22/06/2022 upd 3 Which attribute do you want to update? 1.Task description 2.Duration 3.Due date 4.Category	
Invalid input for upd command	Your TO DO list: 1. icp_assignment 60 22/06/2022 ICP 22/06/2022 2. idb_db_schema 30 22/06/2022 IDB 22/06/2022 3. BMK_video_presentation 30 30/06/2022 IDB 22/06/2022 upd 0 Error: Invalid task position	pass

2 as input for upd command attribute	upd 3 Which attribute do you want to update? 1.Task description 2.Duration 3.Due date 4.Category 2 What is the new value? 30 1.To do list 2.Create My day Schedule 3.My Day 1	pass
Invalid input for upd command attribute	Your TO DO list: 1. icp_assignment 60 22/06/2022 ICP 22/06/2022 2. idb_db_schema 30 22/06/2022 IDB 22/06/2022 3. BMK_video_presentation 30 30/06/2022 IDB 22/06/2022 upd 1 Which attribute do you want to update? 1.Task description 2.Duration 3.Due date 4.Category 5 Error: Invalid option	pass
Different datatype for upd command attribute input	upd 1 Which attribute do you want to update? 1.Task description 2.Duration 3.Due date 4.Category owais Error: Invalid option 1.To do list 2.Create My day Schedule 3.My Day Please choose one of the options by inserting a number Process returned 0 (0x0) execution time : 530.719 s Press any key to continue.	pass

srt	Your TO DO list: 1. icp_assignment 60 22/06/2022 ICP 22/06/2022 2. idb_db_schema 30 22/06/2022 IDB 22/06/2022 3. BMK_video_presentation 30 30/06/2022 IDB 22/06/2022 srt Sort by: 1.Task description 2.Duration 3.Due date 4.Category 5.Date created 1	
Invalid task position provided to Srt command	Your TO DO list: Task Format: description duration due_date category date_created 1. icp_assignment 60 22/06/2022 ICP 22/06/2022 2. idb_db_schema 30 22/06/2022 IDB 22/06/2022 3. BMK_video_presentation 30 30/06/2022 IDB 22/06/2022 srt Sort by: 1.Task description 2.Duration 3.Due date 4.Category 5.Date created 6 Error:invalid input	pass
1 provided as task position to srt command	srt Sort by: 1.Task description 2.Duration 3.Due date 4.Category 5.Date created 1 What order? Ascending=1;Descending=0;	
Text value provided as task position to srt command	srt Sort by: 1.Task description 2.Duration 3.Due date 4.Category 5.Date created owais What order? Ascending=1;Descending=0; 1.To do list 2.Create My day Schedule 3.My Day Please choose one of the options by inserting a number	pass

1 provided to order input in srt command	srt Sort by: 1.Task description 2.Duration 3.Due date 4.Category 5.Date created 1 What order? Ascending=1;Descending=0; 1		
Invalid value provided to order input in srt command	srt Sort by: 1.Task description 2.Duration 3.Due date 4.Category 5.Date created 1 What order? Ascending=1;Descending=0; 3 Error:invalid input		pass

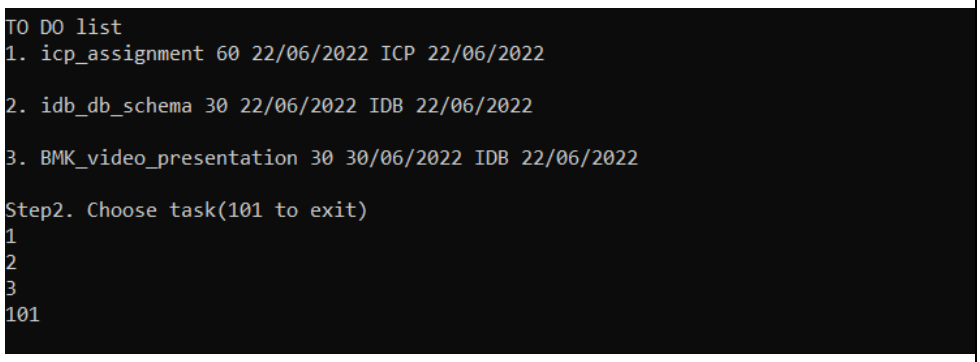
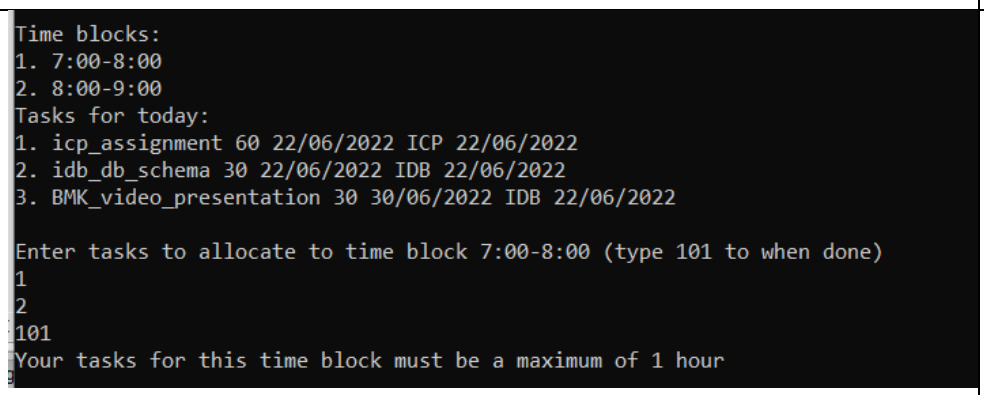
Create my day schedule

Input	Output	Pass/Fail

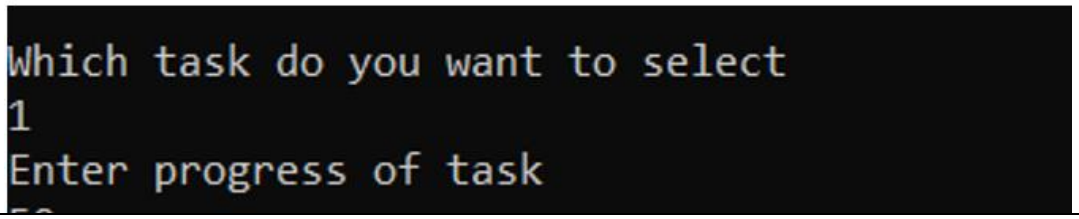
2 provided as input for time block	```Select time block (press 101 when you are done): 2 Is the time rest or work? R/W w Step 1. Time blocks in a day 1 6:00 - 7:00 Rest 2 7:00 - 8:00 Work 3 8:00 - 9:00 Rest 4 9:00 - 10:00 Rest 5 10:00 - 11:00 Rest 6 11:00 - 12:00 Rest 7 12:00 - 13:00 Rest 8 13:00 - 14:00 Rest 9 14:00 - 15:00 Rest 10 15:00 - 16:00 Rest 11 16:00 - 17:00 Rest 12 17:00 - 18:00 Rest 13 18:00 - 19:00 Rest 14 19:00 - 20:00 Rest 15 20:00 - 21:00 Rest 16 21:00 - 22:00 Rest 17 22:00 - 23:00 Rest 18 23:00 - 24:00 Rest 19 24:00 - 1:00 Rest 20 1:00 - 2:00 Rest 21 2:00 - 3:00 Rest 22 3:00 - 4:00 Rest 23 4:00 - 5:00 Rest 24 5:00 - 6:00 Rest Select time block (press 101 when you are done):``` Step 1. Time blocks in a day 1 6:00 - 7:00 Rest 2 7:00 - 8:00 Rest 3 8:00 - 9:00 Rest 4 9:00 - 10:00 Rest 5 10:00 - 11:00 Rest 6 11:00 - 12:00 Rest 7 12:00 - 13:00 Rest 8 13:00 - 14:00 Rest 9 14:00 - 15:00 Rest 10 15:00 - 16:00 Rest 11 16:00 - 17:00 Rest 12 17:00 - 18:00 Rest 13 18:00 - 19:00 Rest 14 19:00 - 20:00 Rest 15 20:00 - 21:00 Rest 16 21:00 - 22:00 Rest 17 22:00 - 23:00 Rest 18 23:00 - 24:00 Rest 19 24:00 - 1:00 Rest 20 1:00 - 2:00 Rest 21 2:00 - 3:00 Rest 22 3:00 - 4:00 Rest 23 4:00 - 5:00 Rest 24 5:00 - 6:00 Rest Select time block (press 101 when you are done): 25 Error:Invalid input Process returned 0 (0x0) execution time : 131.514 s Press any key to continue.		
Invalid time block position provided			pass

W provided as input to rest or work to time block	```Select time block (press 101 when you are done): 2 Is the time rest or work? R/W w Step 1. Time blocks in a day 1 6:00 - 7:00 Rest 2 7:00 - 8:00 Work 3 8:00 - 9:00 Rest 4 9:00 - 10:00 Rest 5 10:00 - 11:00 Rest 6 11:00 - 12:00 Rest 7 12:00 - 13:00 Rest 8 13:00 - 14:00 Rest 9 14:00 - 15:00 Rest 10 15:00 - 16:00 Rest 11 16:00 - 17:00 Rest 12 17:00 - 18:00 Rest 13 18:00 - 19:00 Rest 14 19:00 - 20:00 Rest 15 20:00 - 21:00 Rest 16 21:00 - 22:00 Rest 17 22:00 - 23:00 Rest 18 23:00 - 24:00 Rest 19 24:00 - 1:00 Rest 20 1:00 - 2:00 Rest 21 2:00 - 3:00 Rest 22 3:00 - 4:00 Rest 23 4:00 - 5:00 Rest 24 5:00 - 6:00 Rest Select time block (press 101 when you are done):```	
Invalid input provided to time block type (R/W)	```13 18:00 - 19:00 Rest 14 19:00 - 20:00 Rest 15 20:00 - 21:00 Rest 16 21:00 - 22:00 Rest 17 22:00 - 23:00 Rest 18 23:00 - 24:00 Rest 19 24:00 - 1:00 Rest 20 1:00 - 2:00 Rest 21 2:00 - 3:00 Rest 22 3:00 - 4:00 Rest 23 4:00 - 5:00 Rest 24 5:00 - 6:00 Rest Select time block (press 101 when you are done): 2 Is the time rest or work? R/W N Invalid input```	pass

Invalid task provided to choose task	24 5:00 - 6:00 Rest Your workhours for the day: 2 hours TO DO list 1. icp_assignment 60 22/06/2022 ICP 22/06/2022 2. idb_db_schema 30 22/06/2022 IDB 22/06/2022 3. BMK_video_presentation 30 30/06/2022 IDB 22/06/2022 Step2. Choose task(101 to exit) 4 Error:invalid task Process returned 0 (0x0) execution time : 76.430 s Press any key to continue. (task 4 does not exist)	pass
Text datatype provided to choose task	22 3:00 - 4:00 Rest 23 4:00 - 5:00 Rest 24 5:00 - 6:00 Rest Your workhours for the day: 2 hours TO DO list 1. icp_assignment 60 22/06/2022 ICP 22/06/2022 2. idb_db_schema 30 22/06/2022 IDB 22/06/2022 3. BMK_video_presentation 30 30/06/2022 IDB 22/06/2022 Step2. Choose task(101 to exit) owais Error:invalid task Process returned 0 (0x0) execution time : 14.603 s Press any key to continue.	pass

1,2,3 valid tasks entered for chose tasks		
Entering tasks that exceed one hour for a time block		pass

My day

Input	Output	Pass/Fail
1 entered as task position		

Invalid task provided	My day tasks 1. 8:00-9:00 0% idb_db_schema 30 22/06/2022 IDB 22/06/2022 2. 8:00-9:00 0% BMK_video_presentation 30 30/06/2022 IDB 22/06/2022 Which task do you want to select 3 Error:Task position does not exist 1.To do list 2.Create My day Schedule 3.My Day	pass
Text datatype provided as task	My day tasks 1. 8:00-9:00 0% idb_db_schema 30 22/06/2022 IDB 22/06/2022 2. 8:00-9:00 0% BMK_video_presentation 30 30/06/2022 IDB 22/06/2022 Which task do you want to select owais Error:Task position does not exist 1.To do list 2.Create My day Schedule 3.My Day Please choose one of the options by inserting a number Process returned 0 (0x0) execution time : 128.275 s Press any key to continue.	pass
50 entered as task progress	Enter progress of task 50 Successful 1.To do list 2.Create My day Schedule 3.My Day 3	
Invalid progress	My day tasks 1. 8:00-9:00 0% idb_db_schema 30 22/06/2022 IDB 22/06/2022 2. 8:00-9:00 0% BMK_video_presentation 30 30/06/2022 IDB 22/06/ Which task do you want to select 1 Enter progress of task 101 Progress must be a value between 0 and 100	pass

Conclusion

First and foremost, I would like to thank the lecturer Ms. Mary Ting for teaching me the C language. A satisfactory system has been developed with appropriate validation and functionality. I believe I could have done better in the validation aspect and added more programming concepts however I was not able to do it due to the time constraint. I have learnt much from this assignment and I hope to use this knowledge in my future programming modules.

References

cppreference. (n.d.). *Time_t*. cppreference.com. Retrieved June 22, 2022, from https://en.cppreference.com/w/c/chrono/time_t

C library - . Tutorials Point. (n.d.). Retrieved June 22, 2022, from https://www.tutorialspoint.com/c_standard_library/time_h.htm