PROJECT BCI1093 DATA STRUCTURE & ALGORITHMS

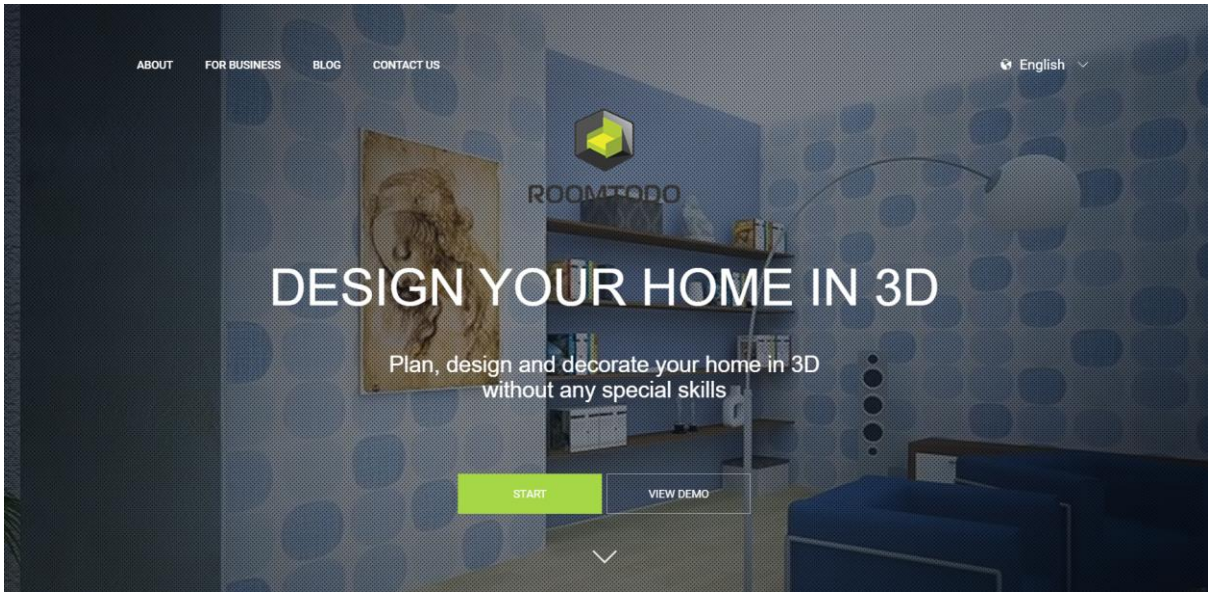
TITLE: VIRTUAL INTERIOR DESIGN AND DECORATION PLATFORM

LECTURER'S NAME: DR. SITI SALWANI YAACOB

MATRIC ID	NAME	SECTION
SD22003	NUR ATIEKA RAFIEKAH BINTI RAZAK	01G
SD22037	NUR NABILA BINTI ABD RAHMAN	01G
SD22011	NOR MIMI AZURA BINTI HUZAIMI	01G
SD22006	SITI MAISARAH BINTI SUHARDI	02G
SD22031	ANIS AQILAH BINTI MOHD ASRI	02G

TABLE OF CONTENTS

CASE STUDY (1): Roomtodo



Roomtodo is an online interior design tool that lets user to plan, design, and decorate house, office, or any other area in 3D without any prior experience. It is a web-based software with a user-friendly interface and powerful capabilities that allows user to design the projects clearly, realistically, and instantly while also giving user with powerful features to design and experiment with interiors.

Roomtodo able to create a 2D floor plan of user apartment, upload it, add windows and doors, experiment with thousands of walls, floor, and ceiling finishes, arrange furniture and decorative objects, and change between various 3D view modes. User may also produce wonderful renders with a single click of a button, transforming user ideas into amazing, realistic images that can carry user project forward.

Roomtodo has a free version that offer just a basic functions and a PRO edition with extra options such as wall material mixing, a baseboard publisher, the ability to upload user's own images, textures, and rugs, and priority support.

The advantages of Roomtodo:

1. Convenient to use:

Roomtodo is a website with an easy-to-use interface and strong features that quickly and realistically help user to visualise their ideas. It also gives user the tools that they need to explore and create their own interiors. The software's user interface makes it simple to visualise the concepts, eliminate challenges, and maximize space allocation.

2. Versatile Function:

Roomtodo allows user to experiment with thousands of walls, floor, and ceiling finishes, arrange furniture and decorative items, make a 2D layout of apartment, upload the apartment plan, add windows and doors, and switch into various 3D view modes. Moreover, user able to explore wonderful colour palettes for perfect décor.

3. Free Software

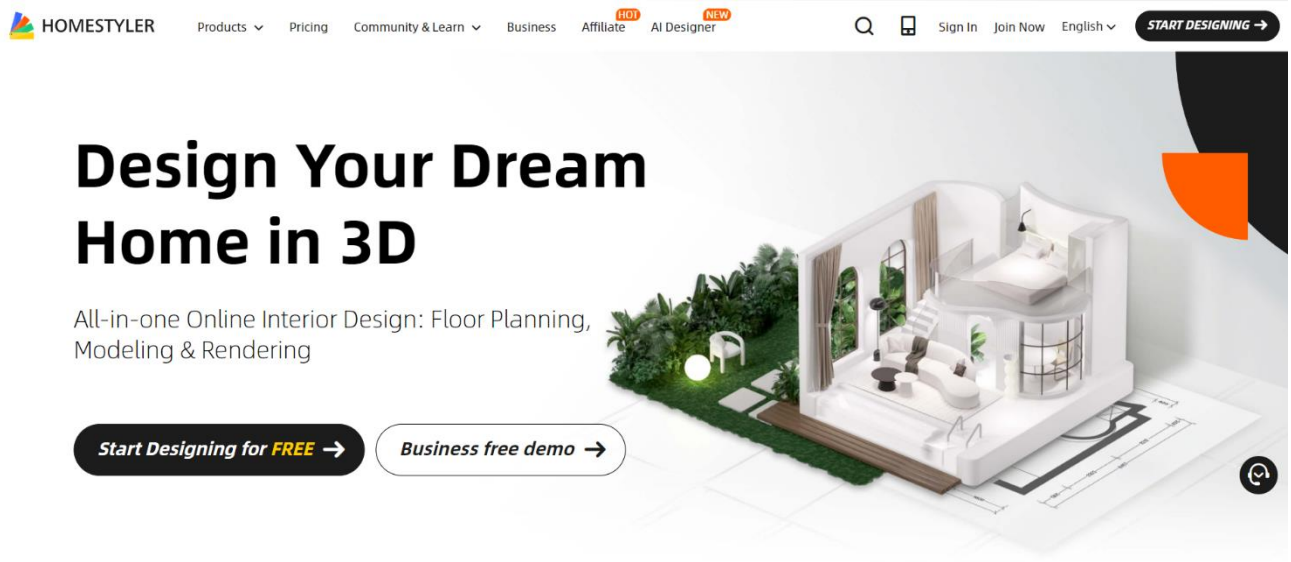
Roomtodo offers an easy and affordable alternative for creating your office or commercial space. With Roomtodo's free layout design tools, user can design a custom office tailored to company's needs without breaking financially.

The disadvantages of Roomtodo:

1. Limited features:
Users may struggle to discover specific design within a virtual design environment due to limited searching tools. This will lead to time consuming because user will spend a lot of time to manually searching through libraries or directories to find certain element. This can be extremely inefficient, especially in large or complicated design projects.

2. Cost consuming:
Roomtodo's PRO edition can be subscribed for $9.99 a month, $19.99 every six months, or $29.99 annually. User must upgrade to the PRO version if want to use the more sophisticated features, such baseboard editing, wall material combination, adding own images, textures, and rugs, and priority help.

CASE STUDY (2): Homestyler



Homestyler is an innovative virtual interior design platform that empowers users to create stunning and personalized interior designs with ease. Developed by Autodesk, this intuitive tool combines simplicity with powerful features, allowing both professionals and enthusiasts to visualize and plan spaces effectively.

The platform offers a user-friendly interface where individuals can experiment with various furniture arrangements, colours, textures, and decor elements within a 3D environment. Homestyler stands out due to its vast library of real furniture and decor items from popular brands, enabling users to accurately simulate their design visions. Moreover, it incorporates augmented reality (AR) capabilities, allowing users to place furniture and designs within their own space through a smartphone or tablet camera, providing a realistic preview of how the design will look in real life.

The platform also facilitates collaboration by enabling users to share their designs, gather feedback, and work collectively on projects. With its blend of functionality and accessibility, Homestyler serves as an invaluable tool for anyone seeking to conceptualize, plan, and visualize interior designs efficiently.

The advantages of Homestyler:

1. Design by Homestyler AI:
With the help of this tool, user may automatically furnish and design space in the style and size that been chosen. All that remains to do is upload a photo of space and AI will take care of the rest.

2. Cross-platform accessibility:
Homestyler is a web-based service that can be accessed on any browser include a mobile and desktop device. User can simply share the projects with others and may be worked on at any time and from any location.

4. Community space:
Homestyler provides a community space where user can show off the ideas to other users and get their opinions and recommendations. Users can find inspiration from their works art with the help of sharing.

The disadvantages of Homestyler:

1.Internet dependency Homestyler:

This website needs to operate online, so designing requires an internet connection. Lack of internet access may hinder the ability to work on projects or access saved designs. This problem may arise when users working on large projects or when there is a lot of traffic on the site.

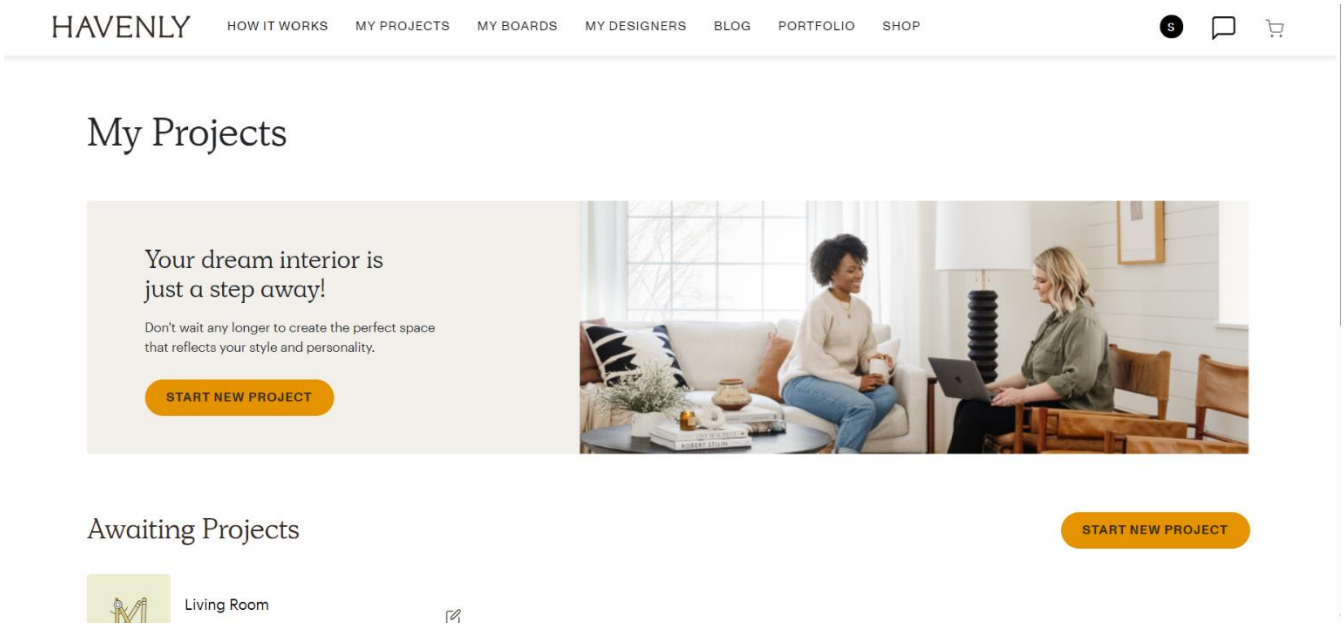
2. Lack of user friendly

Without the option to delete or remove wrong sections, correcting flaws or errors within the design becomes strongly more challenging for the user to fix the mistake. The inability to manipulate and control parts inside their designs may cause users to feel limited because not have flexibility and freedom in controlling their projects.

3. Compatibility Problems:

It may not work properly on some devices or browsers, which could affect functionality or user experience. When features fail to function effectively or the design fails to appear correctly, users may become frustrated, resulting in a negative perception of the product. This may influence on customer satisfaction and discourage involvement.

CASE STUDY (3): Havenly



Havenly is an online resource for interior design that connects users with qualified designers. Users can choose a designer or use recommendation after purchasing one of their packages. The user works with the designer to create a personalised design concept for a selected area. With carefully chosen design options and a customised shopping list, Havenly helps users whether they're looking for finishing touches or a full home makeover.

The assigned designer makes sure that the user's preferences, style, and budget are all taken into consideration in the final design. Before delivering a finalised design concept with links to buy the needed components, they take the time to learn the user's individual tastes. Havenly provides access to online design materials in addition to providing individualised design services.

The advantage of Heavenly:

1. Customized design solutions:

   Havenly's customized design solutions are tailored to user specific needs and preferences. The designer will get to know user own style, preferences, and budget before presenting a finalized design concept with links to purchase the design to select.

.

2. Price guarantee:

   Havenly's price guarantee ensures that use never overpay for their services 1. If user find a lower price for the same service on another website, Havenly will match the price and give you an additional 10% off. This guarantee applies to all Havenly services, including Design Online and Design In-Person

3. Wide range of home brands:

   Havenly's designers source from 100+ home brands so no two spaces are ever alike. Havenly's designers source from over 100 home brands to create unique and personalized designs for each client. This means that no two spaces are ever alike, and user can be sure that their design will be tailored to specific needs and preferences.

The disadvantage of Havenly:

1. No DIY Element

   Havenly's approach lacks a Do-It-Yourself (DIY) element, which means users do not actively participate in the design process. This could be a disadvantage for individuals who enjoy hands-on involvement or want to express their creativity in shaping their living spaces.

2. Limited revisions

   The limited number of design ideas and revisions in the mini package may be restrictive for clients who desire more options or have specific preferences. A constrained design process might lead to dissatisfaction if the final concept doesn't align well with the client's vision.

3. Limited update function

   When a design cannot be easily updated or changed due to limitations in the software or tools, the design process can be complicated for the user. Furthermore, updates or changes that should be simple might lead to disruptions in the workflow, causing delays or complications in project timelines.

CASE STUDY (4): Dreamdesign.co

Dreamdesign.co is a comprehensive virtual design platform that offers services professionals in the architecture and contracting fields. Specializing in 3D visualization, furniture selection, and layout planning, our platform stands out through its user-friendly virtual design process. Users can upload floor plans, choose styles, and engage in collaborative interactions with our expert designers to receive personalized design proposals. The platform boasts a robust set of tools and features, including a diverse library of furniture and decor items, mood board creation tools, virtual walk-throughs, and budgeting capabilities.

The easy search functionality allows users to efficiently find specific elements they desire for their layouts, be it components, styles, furniture, or decorations. The system's flexibility shines through with the ability to remove selected items for cancellation and replace them with updated or alternative choices. This adaptability ensures that customers can fine-tune their designs according to changing preferences or evolving project requirements. Whether searching for specific items, making adjustments, or incorporating new elements, your platform empowers users to effortlessly craft and modify their ideal designs, enhancing the overall convenience and satisfaction of the virtual design process.

Customers have the flexibility to create their own design layouts, with each element contributing to the overall price in Malaysian Ringgit (RM). The Layout section includes predefined spaces like Bedroom, Studio, Kitchen, and Living room, each associated with a specific price point. The Style category offers a range of design aesthetics, including Modern, Contemporary and Minimalist, with associated prices.

What sets Dreamdesign.co apart is its commitment to utilizing cutting-edge technologies for 3D rendering and visualization, ensuring a seamless and immersive experience. We prioritize data security and privacy, employing state-of-the-art measures to safeguard user information and uploaded floor plans. Our platform has earned recognition for its innovative approach, and we proudly showcase case studies

and testimonials from satisfied users. Dreamdesign.co is not just a virtual design service; it's a tailored, collaborative, and secure solution that transforms spaces into dreams.

Layout

Layout
Bedroom
Studio
Kitchen
Living room

Style

Style
Modern
Contemporary
Minimalist

Colour Scheme

Colour
Grey
Blue
White
Brown
Green

Decoration

Decoration	Size
Sofa	1.Small 2.Medium 3.Large
Chair	
Table	
Bed	
Wardrobe	

SCRENSHOTS OF SAMPLE INPUT AND OUTPUT

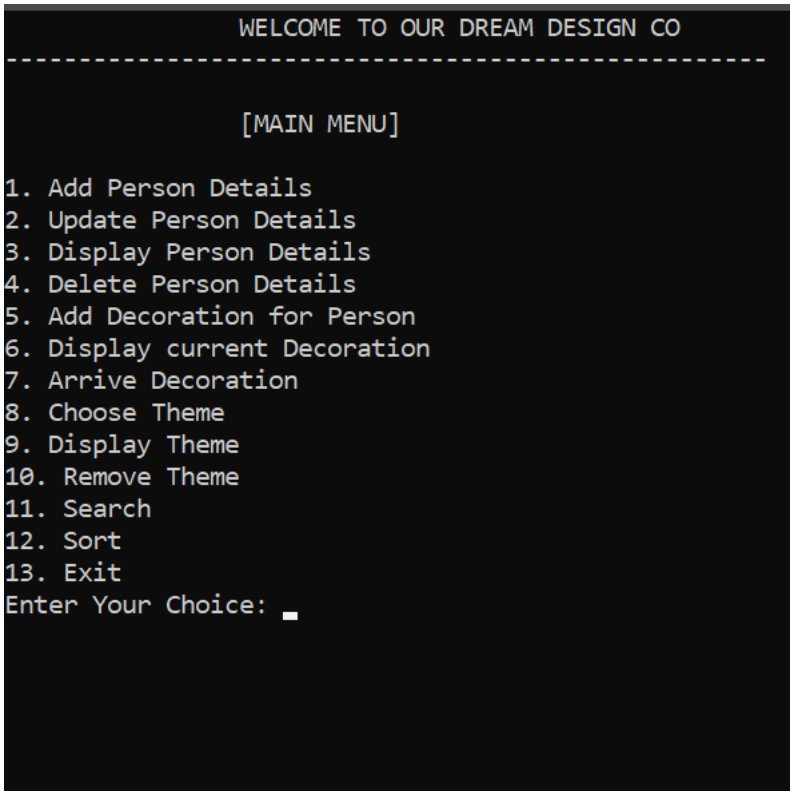


Figure 1 Main Menu

```
PLEASE FILL UP THE PERSON DETAILS BELOW:

Enter name: ATIEKA NABILA

Enter age: 25
----------------
|Layout          |
-----------------
|1. Bedroom      |
|2. Studio       |
|3. Kitchen      |
|4. Living Room  |
----------------

Enter Layout Name: BEDROOM

Enter phone number: 01139475513

Customer Birth Date

Day: 20

Month: 5

Year: 1999

Enter address: KUALA PILAH

Enter budget expenses: 8000

Do you want to continue? (y/n): y
```

Figure 2 Add Person 1 Detail

```
PLEASE FILL UP THE PERSON DETAILS BELOW:

Enter name: MIMI AQILAH

Enter age: 23
-----------------
|Layout          |
-----------------
|1. Bedroom      |
|2. Studio       |
|3. Kitchen      |
|4. Living Room  |
-----------------

Enter Layout Name: STUDIO

Enter phone number: 01111265856

Customer Birth Date

Day: 1

Month: 3

Year: 2001

Enter address: JENGKA

Enter budget expenses: 5000

Do you want to perform another operation? (y/n): Y
```

Figure 3 Add Person 2 Detail

```
PLEASE FILL UP THE PERSON DETAILS BELOW:

Enter name: SITI MAISARA

Enter age: 21
-----------------
|Layout          |
-----------------
|1. Bedroom      |
|2. Studio       |
|3. Kitchen      |
|4. Living Room  |
-----------------

Enter Layout Name: KITCHEN

Enter phone number: 01967545324

Customer Birth Date

Day: 23

Month: 8

Year: 2003

Enter address: SHAH ALAM

Enter budget expenses: 9000

Do you want to continue? (y/n): Y
```

Figure 4 Add Person 3 Detail

```
Enter Your Choice: 2




Enter the name of the person to update: MIMI AQILAH

1. Layout 2. Address
Please enter your choice for update : 1
-----------------
|Layout          |
-----------------
|1. Bedroom      |
|2. Studio       |
|3. Kitchen      |
|4. Living Room  |
-----------------

Enter new layout: LIVING ROOM

Person details updated successfully!

Do you want to continue? (y/n): Y
```

Figure 5 Update Person Detail

```
Name: SITI MAISARA

Age: 21

Customer birthdate: 23/8/2003

Address: SHAH ALAM

Layout: KITCHEN

Phone Number: 01967545324

Budget Expenses: 9000

Name: MIMI AQILAH

Age: 23

Customer birthdate: 1/3/2001

Address: JENGKA

Layout: LIVING ROOM

Phone Number: 01111265856

Budget Expenses: 5000

Name: ATIEKA NABILA

Age: 25

Customer birthdate: 20/5/1999

Address: KUALA PILAH

Layout: BEDROOM

Phone Number: 01139475513

Budget Expenses: 8000

Do you want to continue? (y/n): Y
```

Figure 6 Display Person details.

```
Enter Your Choice: 4



Enter the name of the person to delete: MIMI AQILAH

Person details deleted successfully!

Do you want to continue? (y/n): Y
```

Figure 7 Delete person details.

```
Name: SITI MAISARA

Age: 21

Customer birthdate: 23/8/2003

Address: SHAH ALAM

Layout: KITCHEN

Phone Number: 01967545324

Budget Expenses: 9000


Name: ATIEKA NABILA

Age: 25

Customer birthdate: 20/5/1999

Address: KUALA PILAH

Layout: BEDROOM

Phone Number: 01139475513

Budget Expenses: 8000


Do you want to continue? (y/n): Y
```

Figure 8 Display Person detail.

```
Enter Your Choice: 5



Enter the name of the person to add decoration: SITI MAISARA
-----------------
|Decoration      |
-----------------
|1. Sofa         |
|2. Chair        |
|3. Table        |
|4. Bed          |
|5. Wardrobe     |
-----------------

Enter decoration name:  SOFA

Quantity: 1

Enter size of the decoration (1:Small, 2:Medium, 3:Large): 2

Decoration information added successfully for SITI MAISARA!
```

Figure 9 Add decoration for person.

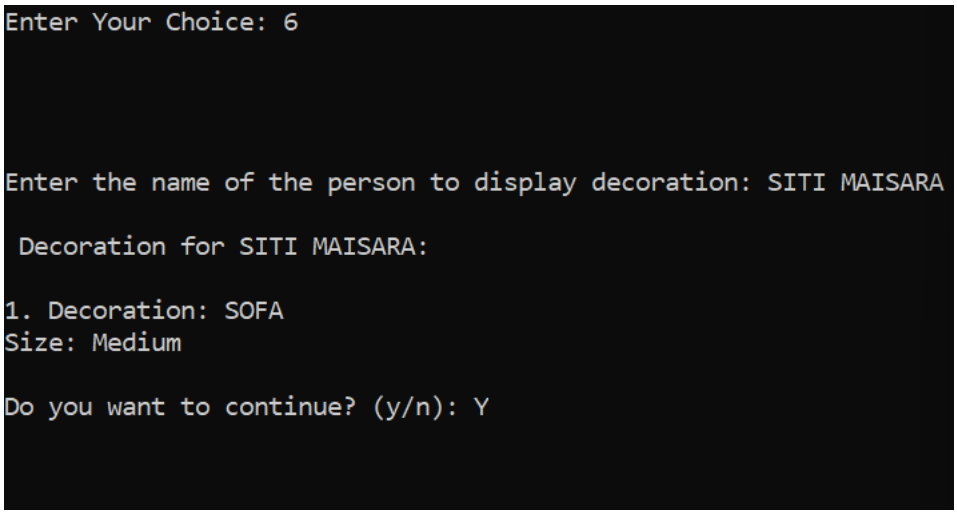


Figure 10 Display current decoration.

```
Enter Your Choice: 7



Enter the name of the person to mark decoration arrive: SITI MAISARA

decoration for SITI MAISARA:

1. Decoration: SOFA
Quantity Decoration: 1
Size of Decoration: 2

Enter the number of the decoration to mark as taken: 1

Decoration marked as arrive for SITI MAISARA!

Do you want to continue? (y/n): Y
```

Figure 11 Decoration arrival status

```
Enter Your Choice: 6



Enter the name of the person to display decoration: SITI MAISARA

No decoration for SITI MAISARA.

Do you want to continue? (y/n): _
```

Figure 12 Display decoration arrival status

```
Enter Your Choice: 8



Enter person's name: ATIEKA NABILA
-----------------
|Colour Scheme   |
-----------------
|1. Grey         |
|2. Blue         |
|3. White        |
|4. Brown        |
|5. Green        |
-----------------

Choose colour: 1
-----------------
|Style           |
-----------------
|1. Modern       |
|2. Minimalist   |
|3. Contemporary |
-----------------

Choose style for the theme: 2

Colour choosen successfully for Minimalist!


Theme display:

Style (1.Modern, 2.Minimalist, 3.Contemporary)

Person: ATIEKA NABILA,
Colour: Grey,
Style: 2

Do you want to continue? (y/n): _
```

Figure 13 Choose theme

```
13. Exit
Enter Your Choice: 9



Theme display:

Style (1.Modern, 2.Minimalist, 3.Contemporary)

Person: ATIEKA NABILA,
Colour: Grey,
Style: 2

Do you want to continue? (y/n):
```

Figure 14 Display theme

```
Enter Your Choice: 10



Front theme dequeued successfully!


The theme queue is empty

Do you want to continue? (y/n):
```

Figure 15 Remove theme

```
Enter Your Choice: 9



The theme queue is empty

Do you want to continue? (y/n):
```

Figure 16 Display remove theme

```
Enter Your Choice: 11



1. Search by Person
2. Search by Layout
 Please enter your choice for searching: 1


Enter the person's name to search: SITI MAISARA

Person found!

Name: SITI MAISARA

Age: 21

Layout: KITCHEN

Phone Number: 01967545324

Do you want to continue? (y/n):
```

Figure 17 Search by person

```
13. Exit
Enter Your Choice: 11



1. Search by Person
2. Search by Layout
 Please enter your choice for searching: 2

Enter the Layout to search (Bedroom, Studio, Kitchen, Living Room): BEDROOM

Person found!

Name: ATIEKA NABILA

Age: 25

 Layout: BEDROOM

Phone Number: 01139475513

Do you want to continue? (y/n):
```

Figure 18 Search by layout

```
Enter Your Choice: 12

1. Sort Person by Budget 2. Sort Person by Alphabet
 Please enter your choice for sorting: 1

Displaying data of all persons:

Name: SITI MAISARA

Age: 21

Customer birthdate: 23/8/2003

Address: SHAH ALAM

Layout: KITCHEN

Phone Number: 01967545324

Budget Expenses: 9000


Name: ATIEKA NABILA

Age: 25

Customer birthdate: 20/5/1999

Address: KUALA PILAH

Layout: BEDROOM

Phone Number: 01139475513

Budget Expenses: 8000


Do you want to continue? (y/n): _
```

Figure 19 Sort person by budget

```
Enter Your Choice: 12

1. Sort Person by Budget 2. Sort Person by Alphabet
 Please enter your choice for sorting: 2

Name: ATIEKA NABILA

Age: 25

Customer birthdate: 20/5/1999

Address: KUALA PILAH

Layout: BEDROOM

Phone Number: 01139475513

Budget Expenses: 8000


Name: SITI MAISARA

Age: 21

Customer birthdate: 23/8/2003

Address: SHAH ALAM

Layout: KITCHEN

Phone Number: 01967545324

Budget Expenses: 9000


Do you want to continue? (y/n):
```

Figure 20 Sort person by alphabet

```
Enter Your Choice: 13



Thank You For using our system !!



Process returned 0 (0x0)   execution time : 903.993 s
Press any key to continue.
```

Figure 21 Exit system

CODING

SOURCE CODE

```
#include <stdio.h>
#include <stdlib.h>
#include<string.h>
#include <ctype.h>


// define struct
struct details
{
      char name[150];
      int age;
      char phoneNum[20];
    char address[200];
    char budget[30];
    char layout[50];
    struct BirthDate * dob;

};
struct BirthDate
{
    int day;
    int month;
    int year;
};
struct Theme
{
    char personName[150];
    char themeName[150];
    int size;
    struct Theme* next;
};

struct Decoration
{
    char personName[150];
    char decoration[50];
    int quantitydecoration;
    int sizedecoration;
```

```
};

struct DecorationStack
{
    struct Decoration data;
    struct DecorationStack* next;
};

struct person
{
      struct details data;
      struct person *ptrnext;
      struct DecorationStack* decorationStack;
};

struct person *headptr,*newptr,*curptr,*prevptr;

struct Theme* front = NULL;
struct Theme* rear = NULL;

struct person *merge(struct person *left, struct person *right);
struct person *mergeSort(struct person *head);


//Declaration function
void insertDataFromUser();
void displayData(struct person *start);
void updatePersonDetail(char name[]);
void searchByName(const char *targetName);
void searchByLayout(const char *targetlayout);
void deletePerson(char name[]);
void addDecoration(struct person* personPtr);
void displayDecorationByName(struct person* personPtr);
void markDecorationTaken(struct person* personPtr);
void chooseTheme(char name[]);
void dequeueTheme();
void displayTheme();
void bubbleSort() ;


//main menu
```

```
int main()
{
    headptr = NULL;

    int choice, optionsearch, optionsort;
    char addMore;

    printf("\t\tWELCOME TO OUR DREAM DESIGN CO  \n");
    printf("-----------------------------------------------------\n");

      do
      {
            printf("\n");
            printf("\t\t[MAIN MENU]");
            printf("\n");
            printf("\n1. Add Person Details ");
            printf("\n2. Update Person Details ");
            printf("\n3. Display Person Details ");
            printf("\n4. Delete Person Details ");
            printf("\n5. Add Decoration for Person ");
            printf("\n6. Display current Decoration  ");
            printf("\n7. Arrive Decoration ");
            printf("\n8. Choose Theme ");
            printf("\n9. Display Theme ");
            printf("\n10. Remove Theme ");
            printf("\n11. Search  ");
            printf("\n12. Sort ");
            printf("\n13. Exit ");

            printf("\nEnter Your Choice: ");
            scanf("%d",&choice);

            switch (choice) {
            case 1:
                  printf("\n\n\n");
                printf("\nPLEASE FILL UP THE PERSON DETAILS BELOW: \n");
                insertDataFromUser();
                break;
            case 2:
                  printf("\n\n\n");
               if (headptr != NULL)
```

```
                    {
                char nameToUpdate[150];
                printf("\nEnter the name of the person to update: ");
                getchar();
                fgets(nameToUpdate, sizeof(nameToUpdate), stdin);
                strtok(nameToUpdate, "\n");
                updatePersonDetail(nameToUpdate);
                        }
                        else
                        {
                printf("\nNo data to update. Please insert data first.\n");
                }
                break;
            case 3:
                  printf("\n\n\n");
                if (headptr != NULL) {
                    displayData(headptr);
                } else {
                    printf("\nNo data to display. Please insert data
first.\n");
                }
                break;
            case 4:
                  printf("\n\n\n");
                if (headptr != NULL) {
                    char nameToDelete[150];
                    printf("\nEnter the name of the person to delete: ");
                    getchar();
                    fgets(nameToDelete, sizeof(nameToDelete), stdin);
                    strtok(nameToDelete, "\n");
                    deletePerson(nameToDelete);
                } else {
                    printf("\nNo data to delete. Please insert data
first.\n");
                }
                break;
            case 5:
                printf("\n\n\n");
                if (headptr != NULL) {
                    char nameToAddDecoration[150];
```

```
                    printf("\nEnter the name of the person to add
decoration: ");
                    getchar();
                    fgets(nameToAddDecoration, sizeof(nameToAddDecoration),
stdin);
                    strtok(nameToAddDecoration, "\n");
                    curptr = headptr;
                    while (curptr != NULL && strcmp(curptr->data.name,
nameToAddDecoration) != 0) {
                        curptr = curptr->ptrnext;
                    }

                    if (curptr != NULL) {
                        addDecoration(curptr);
                    } else {
                        printf("\nPerson with name %s not found.\n",
nameToAddDecoration);
                    }
                } else {
                    printf("\nNo data to add decoration. Please insert data
first.\n");
                }
                  break;
            case 6:
                printf("\n\n\n");
                if (headptr != NULL)
                        {
                    char nameToDisplayDecoration[150];
                    printf("\nEnter the name of the person to display
decoration: ");
                    getchar();
                    fgets(nameToDisplayDecoration,
sizeof(nameToDisplayDecoration), stdin);
                    strtok(nameToDisplayDecoration, "\n");

                    curptr = headptr;
                    while (curptr != NULL && strcmp(curptr->data.name,
nameToDisplayDecoration) != 0) {
                        curptr = curptr->ptrnext;
                    }
```

```
                    if (curptr != NULL) {
                        displayDecorationByName(curptr);
                    } else {
                        printf("\nPerson with name %s not found.\n",
nameToDisplayDecoration);
                    }
                } else {
                    printf("\nNo data to display decoration. Please insert
data first.\n");
                }
                  break;
            case 7:
                printf("\n\n\n");
                if (headptr != NULL) {
                    char nameToMarkDecoration[150];
                    printf("\nEnter the name of the person to mark
decoration arrive: ");
                    getchar();
                    fgets(nameToMarkDecoration,
sizeof(nameToMarkDecoration), stdin);
                    strtok(nameToMarkDecoration, "\n");
                    curptr = headptr;
                    while (curptr != NULL && strcmp(curptr->data.name,
nameToMarkDecoration) != 0) {
                        curptr = curptr->ptrnext;
                    }

                    if (curptr != NULL) {
                        markDecorationTaken(curptr);
                    } else {
                        printf("\nPerson with name %s not found.\n",
nameToMarkDecoration);
                    }
                } else {
                    printf("\nNo data to mark decoration arrive. Please
insert data first.\n");
                }
                  break;
            case 8:
                  printf("\n\n\n");
                  if (headptr != NULL)
```

```
                {
                      char nametoTheme[150];
                      printf("\nEnter person's name: ");
                      getchar();
                  fgets(nametoTheme, sizeof(nametoTheme), stdin);
                  strtok(nametoTheme, "\n");
                  chooseTheme(nametoTheme);
               }
                  else
                      {
                  printf("\nNo data to choose Theme. Please insert data
first.\n");
               }
               break;
            case 9:
                  printf("\n\n\n");
                  displayTheme();
               break;
            case 10:
                  printf("\n\n\n");
                  dequeueTheme();
               break;
            case 11:
                  printf("\n\n\n");
                  printf("1. Search by Person \n2. Search by Layout");
                  printf("\n Please enter your choice for searching: ");
                  scanf("%d", &optionsearch);
                  if (optionsearch==1)
                  {
                        printf("\n");
                        printf("\nEnter the person's name to search: ");
                    char searchName[150];
                    getchar();
                    fgets(searchName, sizeof(searchName), stdin);
                    strtok(searchName, "\n");
                    searchByName(searchName);
                         }
                        else
                        {
                              printf("\nEnter the Layout to search
(Bedroom, Studio, Kitchen, Living Room): ");
```

```
                    char searchLayout[50];
                    getchar();
                    fgets(searchLayout, sizeof(searchLayout), stdin);
                    strtok(searchLayout, "\n");
                    searchByLayout(searchLayout);
                      }
                break;


          case 12:
                printf("\n1. Sort Person by Budget 2. Sort Person by
Alphabet ");
                printf("\n Please enter your choice for sorting: ");
                scanf("%d", &optionsort);
                if (optionsort==1)
                {
                      bubbleSort();
                      }
                      else
                      {
                             if (headptr != NULL)
                            {
                        headptr = mergeSort(headptr);
                        displayData(headptr);
                    }
                            else {
                        printf("\nNo data to sort. Please insert data
first.\n");
                    }
                      }
                break;
          case 13:
                printf("\n\n\n");
                    printf("Thank You For using our system !! ");
                    printf("\n\n\n");
                    exit(0);
              break;
          default:
              printf("\n\t\t               Invalid choice\n");
              break;
       }
```

```
        // Ask the user if they want to continue
        if (choice != 15) {
            printf("\nDo you want to continue? (y/n): ");
            scanf(" %c", &addMore);
            getchar();
        }

    } while (choice != 15 && (addMore == 'y' || addMore == 'Y'));


            if(addMore=='N'||addMore=='n')
            {
                  system("cls");
            printf("\n\n\n");
                printf("\t\t\t\t                      Dream Design Co \n");
                printf("\t\t\t\t          ---------------------------------
--------\n");
                  printf("\n\n\t\t\t\t             Thank You For using our
system !! ");
                  printf("\n\n\n");
                  exit(0);
            }

    curptr = headptr;
    while (curptr != NULL) {
        prevptr = curptr;
        curptr = curptr->ptrnext;
        free(prevptr);
    }

return 0;
}

// Function to add person at the begining//Stack (last-in,first-out)
(LIFO)//push
void insertDataFromUser()
{


    newptr = (struct person *)malloc(sizeof(struct person));
```

```
if (newptr == NULL) {
    printf("Memory allocation failed.\n");
    exit(1);
}

newptr->data.dob = (struct BirthDate*)malloc(sizeof(struct BirthDate));
  if (newptr->data.dob == NULL) {
    printf("Memory allocation failed.\n");
    exit(1);
}



printf("\nEnter name: ");
getchar();
fgets(newptr->data.name, sizeof(newptr->data.name), stdin);
strtok(newptr->data.name, "\n");

printf("\nEnter age: ");
scanf("%d", &newptr->data.age);

printf("-----------------\n");
printf("|Layout          |\n");
printf("-----------------\n");
printf("|1. Bedroom      |\n");
printf("|2. Studio       |\n");
printf("|3. Kitchen      |\n");
printf("|4. Living Room  |\n");
printf("-----------------\n");
  printf("\nEnter Layout Name: ");
getchar();
fgets(newptr->data.layout, sizeof(newptr->data.layout), stdin);
strtok(newptr->data.layout, "\n");

printf("\nEnter phone number: ");
fgets(newptr->data.phoneNum, sizeof(newptr->data.phoneNum), stdin);
strtok(newptr->data.phoneNum, "\n");

printf("\nCustomer Birth Date\n");
printf("\nDay: ");
  scanf("%d", &newptr->data.dob->day);
```

```
      printf("\nMonth: ");
      scanf("%d", &newptr->data.dob->month);

      printf("\nYear: ");
      scanf("%d", &newptr->data.dob->year);

      fflush (stdin);
    printf("\nEnter address: ");
    fgets(newptr->data.address, sizeof(newptr->data.address), stdin);
    strtok(newptr->data.address, "\n");



    printf("\nEnter budget expenses: ");
    fgets(newptr->data.budget, sizeof(newptr->data.budget), stdin);
    strtok(newptr->data.budget, "\n");

    newptr->decorationStack = NULL;

    newptr->ptrnext = headptr;
    headptr = newptr;
}


// Function to display data of each person in the linked list
void displayData(struct person *start)
{
    curptr = start;

    // Traverse the linked list and print each person's data
    while (curptr != NULL) {
        printf("\nName: %s\n", curptr->data.name);
        printf("\nAge: %d\n", curptr->data.age);
        printf("\nCustomer birthdate: %d/%d/%d\n", curptr->data.dob->day,
curptr->data.dob->month, curptr->data.dob->year );
        printf("\nAddress: %s\n", curptr->data.address);
        printf("\nLayout: %s\n", curptr->data.layout);
        printf("\nPhone Number: %s\n", curptr->data.phoneNum);
        printf("\nBudget Expenses: %s\n", curptr->data.budget);
        printf("\n");

        curptr = curptr->ptrnext;
```

```
    }
}

// Function to update a specific detail of a person based on their name
void updatePersonDetail(char name[])
{
    curptr = headptr;

    while (curptr != NULL && strcmp(curptr->data.name, name) != 0) {
        prevptr = curptr;
        curptr = curptr->ptrnext;
    }

    if (curptr != NULL) {
        printf("\n1. Layout 2. Address");
        printf("\nPlease enter your choice for update : ");

        int detailChoice;
        scanf("%d", &detailChoice);

        getchar();

        switch (detailChoice) {
            case 1:
                printf("-----------------\n");
                printf("|Layout          |\n");
                printf("-----------------\n");
                printf("|1. Bedroom      |\n");
                printf("|2. Studio       |\n");
                printf("|3. Kitchen      |\n");
                printf("|4. Living Room  |\n");
                printf("-----------------\n");
                printf("\nEnter new layout: ");
                fgets(curptr->data.layout, sizeof(curptr->data.layout),
stdin);
                strtok(curptr->data.layout, "\n");
                break;
            case 2:
                // Update address
                printf("\nEnter new Address: ");
```

```
                fgets(curptr->data.address, sizeof(curptr->data.address),
stdin);
                strtok(curptr->data.address, "\n");
                break;
            default:
                printf("\nInvalid choice. No details updated.\n");
                return;
        }

        printf("\nPerson details updated successfully!\n");
    } else {
        printf("\nPerson not found.\n");
    }
}

//Implement searching person using linear sequential searching by name
void searchByName(const char *targetName)
{
    curptr = headptr;
    int found = 0;

    while (curptr != NULL) {
        if (strcmp(curptr->data.name, targetName) == 0) {
            found = 1;
            break;
        }
        curptr = curptr->ptrnext;
    }

    if (found) {
        printf("\nPerson found!\n");
        printf("\nName: %s\n", curptr->data.name);
        printf("\nAge: %d\n", curptr->data.age);
        printf("\nLayout: %s\n", curptr->data.layout);
        printf("\nPhone Number: %s\n", curptr->data.phoneNum);
    } else {
        printf("\nPerson not found.\n");
    }
}

//Implement searching using linear sequential searching by layout
```

```
void searchByLayout(const char *targetLayout)
{
    curptr = headptr;
    int found = 0;

    while (curptr != NULL) {
        if (strcmp(curptr->data.layout, targetLayout) == 0) {
            found = 1;
            break;
        }
        curptr = curptr->ptrnext;
    }

    if (found) {
        printf("\n\Person found!\n");
        printf("\nName: %s\n", curptr->data.name);
        printf("\nAge: %d\n", curptr->data.age);
        printf("\n Layout: %s\n", curptr->data.layout);
        printf("\n\Phone Number: %s\n", curptr->data.phoneNum);
    } else {
        printf("\nPerson not found.\n");
    }
}


//implement linked list for delete by person name from the list
void deletePerson(char name[])
{
    curptr = headptr;
    struct person *temp;

    if (curptr != NULL && strcmp(curptr->data.name, name) == 0) {
        headptr = curptr->ptrnext;
        free(curptr);
        printf("\nPerson details deleted successfully!\n");
        return;
    }

    // Search for the person to delete
    while (curptr != NULL && strcmp(curptr->data.name, name) != 0) {
        prevptr = curptr;
```

```
        curptr = curptr->ptrnext;
    }

    // If the person is found, delete them
    if (curptr != NULL) {
        prevptr->ptrnext = curptr->ptrnext;
        free(curptr);
        printf("\nPerson details deleted successfully!\n");
    } else {
        printf("\n\Person not found.\n");
    }
}

//implement linked list for display detail from the list
void displaydetails()
{
    printf("\nDisplaying data of all persons:\n");
    displayData(headptr);
}

// Modify the function definition for the addDecoration function
void addDecoration(struct person* personPtr)
{
    struct Decoration newDecoration;
    printf("-----------------\n");
    printf("|Decoration      |\n");
    printf("-----------------\n");
    printf("|1. Sofa         |\n");
    printf("|2. Chair        |\n");
    printf("|3. Table        |\n");
    printf("|4. Bed          |\n");
    printf("|5. Wardrobe     |\n");
    printf("-----------------\n");
    printf("\nEnter decoration name: ");
    getchar();
    fgets(newDecoration.decoration, sizeof(newDecoration.decoration),
stdin);
    strtok(newDecoration.decoration, "\n");

    printf("\nQuantity: ");
    scanf("%d", &newDecoration.quantitydecoration);
```

```
    if (newDecoration.quantitydecoration > 1)
       {
        for (int i = 0; i < newDecoration.quantitydecoration; ++i) {
            printf("\nEnter size for decoration %d (1:Small, 2:Medium,
3:Large): ", i + 1);
            scanf("%d", &newDecoration.sizedecoration);

            // Create a new node for the Decoration
            struct DecorationStack* newNode = (struct
DecorationStack*)malloc(sizeof(struct DecorationStack));

            if (newNode == NULL) {
                printf("\nMemory allocation failed!\n");
                exit(1);
            }

            newNode->data = newDecoration;
            newNode->next = personPtr->decorationStack;
            personPtr->decorationStack = newNode;
        }
    } else {
        printf("\nEnter size of the decoration (1:Small, 2:Medium,
3:Large): ");
        scanf("%d", &newDecoration.sizedecoration);

        struct DecorationStack* newNode = (struct
DecorationStack*)malloc(sizeof(struct DecorationStack));

        if (newNode == NULL) {
            printf("\n\nMemory allocation failed!\n");
            exit(1);
        }

        newNode->data = newDecoration;
        newNode->next = personPtr->decorationStack;
        personPtr->decorationStack = newNode;
    }

    printf("\nDecoration information added successfully for %s!\n",
personPtr->data.name);
```

```
}


//display function for decoration by name
void displayDecorationByName(struct person* personPtr)
{
    if (personPtr->decorationStack == NULL)
    {
        printf("\nNo decoration for %s.\n", personPtr->data.name);
        return;
    }

    printf("\n Decoration for %s:\n", personPtr->data.name);

    struct DecorationStack* currentDecoration = personPtr->decorationStack;
    int count = 1;

    while (currentDecoration != NULL) {
        printf("\n%d. Decoration: %s\n", count, currentDecoration-
>data.decoration);

        if (currentDecoration->data.sizedecoration == 1)
        {
            printf("Size: Small\n");
        }
        else if (currentDecoration->data.sizedecoration == 2)
        {
            printf("Size: Medium\n");
        }
        else
        {
            printf("Size: Large\n");
        }

        currentDecoration = currentDecoration->next;
        count++;
    }
}


//Display status of decoration arrival using stack//pop
```

```
void markDecorationTaken(struct person* personPtr)
{
    if (personPtr->decorationStack == NULL) {
        printf("\n No decoration marks to arrive for %s.\n", personPtr-
>data.name);
        return;
    }

    printf("\ndecoration for %s:\n", personPtr->data.name);

    struct DecorationStack* currentDecoration = personPtr->decorationStack;
    int count = 1;

    // Display decoration and let the user choose which one to mark as
arived
    while (currentDecoration != NULL) {
        printf("\n%d. Decoration: %s\n", count, currentDecoration-
>data.decoration);
        printf("Quantity Decoration: %d\n", currentDecoration-
>data.quantitydecoration);
        printf("Size of Decoration: %d\n", currentDecoration-
>data.sizedecoration);

        currentDecoration = currentDecoration->next;
        count++;
    }

    int choice;
    printf("\nEnter the number of the decoration to mark as taken: ");
    scanf("%d", &choice);

    // Remove the chosen decoration node from the stack
    if (choice == 1) {
        // Special case if the first node is chosen
        struct DecorationStack* temp = personPtr->decorationStack;
        personPtr->decorationStack = temp->next;
        free(temp);
    } else {
        currentDecoration = personPtr->decorationStack;
        for (int i = 1; i < choice - 1 && currentDecoration->next != NULL;
i++) {
```

```
            currentDecoration = currentDecoration->next;
        }

        if (currentDecoration->next != NULL) {
            struct  DecorationStack* temp = currentDecoration->next;
            currentDecoration->next = temp->next;
            free(temp);
        } else {
            printf("\nInvalid choice. No decoration removed.\n");
            return;
        }
    }

    printf("\nDecoration marked as arrive for %s!\n", personPtr-
>data.name);
}

//choose a new theme for person to choose colour scheme
void chooseTheme(char name[])
{
      curptr = headptr;
    int personFound = 0;

    while (curptr != NULL) {
        if (strcmp(curptr->data.name, name) == 0) {
            personFound = 1;
            break;
        }
        curptr = curptr->ptrnext;
    }

    if (!personFound) {
        printf("\nPerson with name %s not found in the system.\n", name);
        return;
    }

    struct Theme newTheme;
    strcpy(newTheme.personName, name);

    // Choose colour scheme
    printf("-----------------\n");
```

```
    printf("|Colour Scheme   |\n");
    printf("-----------------\n");
    printf("|1. Grey         |\n");
    printf("|2. Blue         |\n");
    printf("|3. White        |\n");
    printf("|4. Brown        |\n");
    printf("|5. Green        |\n");
    printf("-----------------\n");
    printf("\nChoose colour: ");
    int themeOption;
    scanf("%d", &themeOption);

    switch (themeOption) {
        case 1:
            strcpy(newTheme.themeName, "Grey");
            break;
        case 2:
            strcpy(newTheme.themeName, "Blue");
            break;
        case 3:
            strcpy(newTheme.themeName, "White");
            break;
        case 4:
            strcpy(newTheme.themeName, "Brown");
            break;
        case 5:
            strcpy(newTheme.themeName, "Green");
            break;
        default:
            printf("\nInvalid colour. Please select 1 to 5.\n");
            return;
    }

    // Choose style
    printf("-----------------\n");
    printf("|Style           |\n");
    printf("-----------------\n");
    printf("|1. Modern       |\n");
    printf("|2. Minimalist   |\n");
    printf("|3. Contemporary |\n");
    printf("-----------------\n");
```

```
    printf("\nChoose style for the theme: ");
    scanf("%d", &newTheme.size);

    if (newTheme.size < 1 || newTheme.size > 3) {
        printf("\nInvalid theme. Please select 1, 2, or 3.\n");
        return;
    }

    printf("\nColour choosen successfully for ");

    switch (newTheme.size) {
        case 1:
            printf("Modern");
            break;
        case 2:
            printf("Minimalist");
            break;
        case 3:
            printf("Contemporary");
            break;
    }

    printf("!\n");

    struct Theme* newNode = (struct Theme*)malloc(sizeof(struct Theme));

    if (newNode == NULL) {
        printf("\n\nMemory allocation failed!\n");
        return;
    }

    strcpy(newNode->personName, newTheme.personName);
    strcpy(newNode->themeName, newTheme.themeName);
    newNode->size= newTheme.size;
    newNode->next = NULL;

    //enque
    //check if queue is empty

    if (front == NULL) {
        front = rear = newNode;
```

```
    } else {
        rear->next = newNode; //rear point to new node
        rear = newNode; //assign new node as rear
    }

    // Display the updated list of theme
    displayTheme();
}

//implement deque operation using linked list
void dequeueTheme()
{
    if (front == NULL) {
        printf("\n\n!!! Theme Queue is EMPTY - Cannot dequeue !!!\n");
    } else {
        struct Theme* temp = front;
        front = front->next;
        free(temp);
        printf("\n\nFront theme dequeued successfully!\n");
    }

    displayTheme();
}

//display theme using linked list
void displayTheme()
{
    if (front == NULL) {
        printf("\n\nThe theme queue is empty\n");
    } else {
        struct Theme* current = front;

        printf("\n\nTheme display:\n");
        printf("\nStyle (1.Modern, 2.Minimalist, 3.Contemporary)\n");
        while (current != NULL) {
            printf("\nPerson: %s, \nColour: %s, \nStyle: %d\n", current-
>personName, current->themeName, current->size);
            current = current->next;
        }
    }
}
```

```
//implement sorting using bubble sort by budget
void bubbleSort()
{
       if (headptr == NULL || headptr->ptrnext == NULL) {
        // No need to sort if the list is empty or has only one element
        return;
    }

    int swapped;
    struct person *last = NULL;

    do {
        swapped = 0;
        curptr = headptr;

        while (curptr->ptrnext != last) {
            if (curptr->data.age > curptr->ptrnext->data.age) {
                // Swap nodes
                struct person *temp = curptr->ptrnext;
                curptr->ptrnext = temp->ptrnext;
                temp->ptrnext = curptr;
                if (curptr == headptr) {
                    headptr = temp;
                } else {
                    prevptr->ptrnext = temp;
                }
                swapped = 1;
            } else {
                prevptr = curptr;
                curptr = curptr->ptrnext;
            }
        }

        last = curptr;

    } while (swapped);

    displaydetails();

}
```

```
// implement sorting using merge sort by name alphabatical
struct person *merge(struct person *left, struct person *right)
{
      struct person *result = NULL;

    if (left == NULL) {
        return right;
    }
    if (right == NULL) {
        return left;
    }

    if (strcmp(left->data.name, right->data.name) <= 0) {
        result = left;
        result->ptrnext = merge(left->ptrnext, right);
    } else {
        result = right;
        result->ptrnext = merge(left, right->ptrnext);
    }

    return result;
}

struct person *mergeSort(struct person *head)
{
      if (head == NULL || head->ptrnext == NULL) {
        return head;
    }

    struct person *slow = head;
    struct person *fast = head->ptrnext;

    while (fast != NULL && fast->ptrnext != NULL) {
        slow = slow->ptrnext;
        fast = fast->ptrnext->ptrnext;
    }

    struct person *left = head;
    struct person *right = slow->ptrnext;
    slow->ptrnext = NULL;
```

```
    left = mergeSort(left);
    right = mergeSort(right);

    return merge(left, right);

    displaydetails();
}
```

DISTRIBUTION OF TASKS

ELEMENT	DESCRIPTION	PERSON IN CHARGE
Overall Program	Appropriate usage of variables declarations	NUR NABILA BINTI ABD RAHMAN
	Appropriate usage of comments and indentation	
	Error-free program	
Structure	Main Structure declaration (minimum 2)	NUR ATIEKA RAFIEKAH BINTI RAZAK
	Nested Structure declaration (minimum 2)	
	Implement passing structure to a function (minimum 2)	
	Implement return a structure to a calling function (minimum 2)	
	Able to create, assign and access structures variables/elements (minimum 15)	
Linked List	Able to insert data into the list	NOR MIMI AZURA BINTI HUZAIMI
	Able to delete data from the list	
	Able to modify / update data in the list	
Stack	Able to implement Push operation using Linked List / Array	SITI MAISARAH BINTI SUHARDI
	Able to implement Pop operation using Linked List / Array	
Searching	Implement searching using any of the available search methods (Linear Sequential Search, Ordered Sequential Search, Binary Search) User able to search and retrieve the required information	ANIS AQILAH BINTI MOHD ASRI
Sorting	Implement sorting using any of the available sorting methods (Selection,	NUR NABILA BINTI ABD RAHMAN

	Insertion, Bubble, Quick, Merge)	
	The user is given the option to select the category/item to be sorted	
	Users can view the sorted list based on the selected category/item	
Additional Information / Case Study (10m)	Able to find additional information/details for the selected title/case study (Minimum 3 related case studies)	SITI MAISARAH BINTI SUHARDI
	Provide online citations/links/sources	
Main Menu	Appropriate design of Menu category/item	NUR ATIEKA RAFIEKAH BINTI RAZAK
	Appropriate arrangement of the Menu options/items	

REFERENCE

Homestyler - Free 3D Home Design Software & Floor Planner Online. (n.d.). Homestyler. https://www.homestyler.com/

Online Interior Design And Home Inspiration | Havenly. (n.d.). Havenly.com. Retrieved January 16, 2024.

https://havenly.com/rooms/

Roomtodo - free interior and decorating home design in 3D Online. (n.d.). Roomtodo.

https://roomtodo.com/en/