****

**Green University of Bangladesh**

**Department of Computer Science and Engineering (CSE)**

**Faculty of Sciences and Engineering**

**Semester: (Spring, Year:2021), B.Sc. in CSE (Day/Eve)**

**Course Title : Data Structures Lab**

**Course Code : CSE 106**

**Section : D2**

**Lab Project Name : Data Structure All Applicaton.**

**Student Details**

| **Name** | | **ID** |
| --- | --- | --- |
| **1.** | Jahidul Islam | 221002504 |
| **2.** |  |  |
| **3.** |  |  |

**Submission Date : 30/12/2022**

**Course Teacher’s Name : Umme Habiba**

**[For Teachers use only: Don’t Write Anything inside this box]**

| **Lab Project Status**  **Marks: ………………………………… Signature: .....................**  **Comments: .............................................. Date: ..............................** |
| --- |

Table of Contents

**Chapter 1 Introduction 3**

1.1 Introduction 3

1.2 Objective 3

**Chapter 2 Design/Development/Implementation of the Project 4**

2.1 Data flow diagram for the current system 4

2.2 Application of Data Sturcture’s Algorithm 4

**Chapter 3 Performance Evaluation 5**

3.1 Simulation Environment/ Simulation Procedure 5

3.2 Results and Discussions 5

**Chapter 4 Conclusion 6**

4.2 Scope of Future Work 6

**References 7**

# Chapter 1 Introduction

## Introduction

Ticket management system, Bookshelf Management system

The student management system allows authorized members to access the record of academically registered students.

## Objective

It can be used in various educational institutes across various educational institutes and simplifies working of institute

# Chapter 2

# Design and Implementation of the Project

## Data flow diagram for the current system

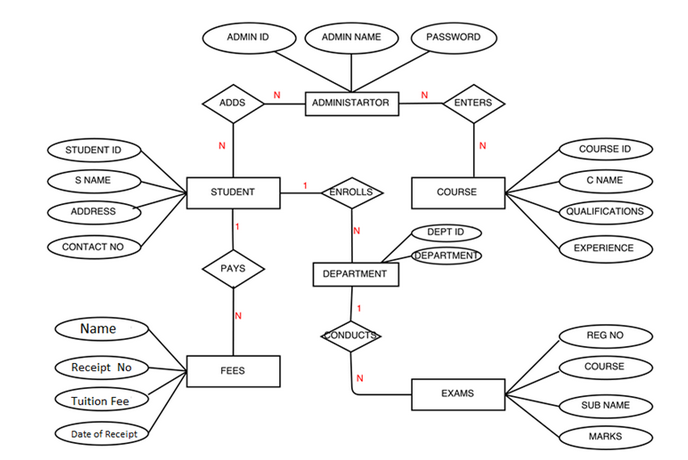


Figure 2.1: Project Diagram of the program.

## 

## 

Code:

#include <stdlib.h>

#include <stdio.h>

#define MAX 10

int front = -1, rear = -1, circQueue[MAX];

int password[10];

int userName[10];

int counter = 1;

void create();

void display();

void insert\_begin();

void insert\_end();

void insert\_pos();

void delete\_begin();

void delete\_end();

void delete\_pos();

int Linkedlist();

struct node

{

int info;

struct node \*next;

};

struct node \*start = NULL;

// Log in with lear search

int login()

{

userName[0] = 1;

password[0] = 1;

int kk = 0;

int lg1, lg2;

printf("Enter username and Password:");

scanf("%d %d", &lg1, &lg2);

for (int i = 0; i < 10; i++)

{

if (password[i] == lg1 && userName[i] == lg2)

{

kk++;

}

}

if (kk >= 10)

{

printf(" Wrong\n");

exit(0);

}

}

void enqueue()

{

int num;

printf("Enter the number to be inserted: ");

scanf("%d", &num);

if (rear == MAX - 1)

{

printf("Overflow");

}

else if (front == -1 && rear == -1)

{

front = 0;

rear = 0;

}

else if (rear == MAX - 1 && front != 0)

{

rear = 0;

}

else

{

rear += 1;

}

circQueue[rear] = num;

}

void dequeue()

{

if (front == -1)

{

printf("noone in line");

}

else if (front == rear)

{

front = -1;

rear = -1;

}

else if (front == MAX - 1)

{

front = 0;

}

else

{

front += 1;

}

}

void displays()

{

int i;

if (front == -1 || rear == -1)

{

printf("\nEmpty line\n");

}

else

{

printf("The line is: ");

for (i = front; i <= rear; i++)

{

printf("%d ", circQueue[i]);

}

printf("\n");

}

}

int Qeue()

{

printf("Welcome to ticket manage ment systtem:\n");

int choice;

while (1)

{

printf("Enter your choice: \n1. Join n line forticket \n");

printf("\n2.Ticket provide:\n");

printf("\n3. Display the line of numbers are wating forticket\n4. Exit\n");

printf("4. Exit\n");

scanf("%d", &choice);

switch (choice)

{

case 1:

enqueue();

break;

case 2:

dequeue();

break;

case 3:

displays();

break;

case 4:

return 0;

default:

printf("Enter valid choice");

break;

}

}

return 0;

}

void swap(int \*x, int \*y)

{

int t = \*x;

\*x = \*y;

\*y = t;

}

int Sorting()

{

int size;

printf("ENter the number of student ");

scanf("%d", &size);

int a[5];

int i, j, tem;

int value, hole;

for (int i = 0; i < size; i++)

{

printf("The marks of student %d: \n", i);

scanf("%d", &a[i]);

}

int choice;

while (1)

{

printf("\n Student marking management system \n");

printf("\n 1.Bubble Sort for sortd the result \n");

printf("\n 2.Insertion Sort for sortd the result \n");

printf("\n 3.Selection Sort for sortd the result \n");

printf("Enter your choice:\t");

scanf("%d", &choice);

switch (choice)

{

case 1:

for (i = 1; i < 5; i++)

{

for (j = 0; j < 5 - i; j++)

{

if (a[j] > a[j + 1])

{

tem = a[j];

a[j] = a[j + 1];

a[j + 1] = tem;

}

}

}

printf("The reslt of lowest to highest:");

for (i = 0; i < 5; i++)

{

printf("%d ", a[i]);

}

break;

case 2:

for (int i = 1; i < size; i++)

{

value = a[i];

hole = i;

while (hole > 0 && a[hole - 1] > value)

{

a[hole] = a[hole - 1];

hole--;

}

a[hole] = value;

}

printf("The reslt of lowest to highest:");

for (i = 0; i < size; i++)

{

printf("%d ", a[i]);

}

break;

case 3:

int i, j;

for (i = 0; i < size - 1; i++)

{

for (j = i + 1; j < size; j++)

{

if (a[i] > a[j])

{

swap(&a[i], &a[j]);

}

}

}

printf("The reslt of lowest to highest:");

for (i = 0; i < size; i++)

{

printf("%d ", a[i]);

}

return 0;

break;

default:

printf("\n Wrong Choice:\n");

break;

}

}

return 0;

}

void create\_linkedList()

{

struct node \*temp, \*ptr;

temp = (struct node \*)malloc(sizeof(struct node));

if (temp == NULL)

{

printf("\nOut of shelf Space:\n");

exit(0);

}

printf("\nEnter the page of book for the Cell:\t");

scanf("%d", &temp->info);

temp->next = NULL;

if (start == NULL)

{

start = temp;

}

else

{

ptr = start;

while (ptr->next != NULL)

{

ptr = ptr->next;

}

ptr->next = temp;

}

}

void display\_linkedList()

{

struct node \*ptr;

if (start == NULL)

{

printf("\nBook self is empty:\n");

return;

}

else

{

ptr = start;

printf("\nThe book List are:\n");

while (ptr != NULL)

{

printf("%d\t", ptr->info);

ptr = ptr->next;

}

}

}

void insertBegin\_linkedList()

{

struct node \*temp;

temp = (struct node \*)malloc(sizeof(struct node));

if (temp == NULL)

{

printf("\nsorry not found the Space:\n");

return;

}

printf("\nEnter the page of book for the Cell\t");

scanf("%d", &temp->info);

temp->next = NULL;

if (start == NULL)

{

start = temp;

}

else

{

temp->next = start;

start = temp;

}

}

void insertEnd\_linkedList()

{

struct node \*temp, \*ptr;

temp = (struct node \*)malloc(sizeof(struct node));

if (temp == NULL)

{

printf("\nsorry not found the Space:\n");

return;

}

printf("\nEnter the page of book for the Cell:\t");

scanf("%d", &temp->info);

temp->next = NULL;

if (start == NULL)

{

start = temp;

}

else

{

ptr = start;

while (ptr->next != NULL)

{

ptr = ptr->next;

}

ptr->next = temp;

}

}

void insertAnyPosition\_linkedList()

{

struct node \*ptr, \*temp;

int i, pos;

temp = (struct node \*)malloc(sizeof(struct node));

if (temp == NULL)

{

printf("\nsorry not found the Space:\n");

return;

}

printf("\nEnter the position for the new book to be inserted:\t");

scanf("%d", &pos);

printf("\nEnter the page number of the node:\t");

scanf("%d", &temp->info);

temp->next = NULL;

if (pos == 0)

{

temp->next = start;

start = temp;

}

else

{

for (i = 0, ptr = start; i < pos - 1; i++)

{

ptr = ptr->next;

if (ptr == NULL)

{

printf("\nPosition not found for you\n");

return;

}

}

temp->next = ptr->next;

ptr->next = temp;

}

}

void deleteAtBeginning\_linkedList()

{

struct node \*ptr;

if (ptr == NULL)

{

printf("\ncell is Empty:\n");

return;

}

else

{

ptr = start;

start = start->next;

printf("\nThe remove book is :%d\t", ptr->info);

free(ptr);

}

}

void deleteAtEnd\_linkedList()

{

struct node \*temp, \*ptr;

if (start == NULL)

{

printf("\nCell is Empty:");

exit(0);

}

else if (start->next == NULL)

{

ptr = start;

start = NULL;

printf("\nThe remove book is:%d\t", ptr->info);

free(ptr);

}

else

{

ptr = start;

while (ptr->next != NULL)

{

temp = ptr;

ptr = ptr->next;

}

temp->next = NULL;

printf("\nThe remove Book is:%d\t", ptr->info);

free(ptr);

}

}

void deleteAtPosition\_linkedList()

{

int i, pos;

struct node \*temp, \*ptr;

if (start == NULL)

{

printf("\nThe book cell is Empty:\n");

exit(0);

}

else

{

printf("\nEnter the position of the book to be deleted:\t");

scanf("%d", &pos);

if (pos == 0)

{

ptr = start;

start = start->next;

printf("\nThe remove book is:%d\t", ptr->info);

free(ptr);

}

else

{

ptr = start;

for (i = 0; i < pos; i++)

{

temp = ptr;

ptr = ptr->next;

if (ptr == NULL)

{

printf("\nPosition not Found:\n");

return;

}

}

temp->next = ptr->next;

printf("\nThe remove book is:%d\t", ptr->info);

free(ptr);

}

}

}

int Stack()

{

int stack[10];

int top = -1;

int max;

int d;

printf("Welcome to flower cercit programing system\n\n");

printf("Enter your Racket flowers number:");

scanf("%d", &max);

while (1)

{

printf("\nEnter 1 to add\n");

printf("Enter 2 to remove\n");

printf("Enter 3 for output\n");

printf("Enter 4 for exit\n");

scanf("%d", &d);

switch (d)

{

case 1:

if (top == max)

{

printf("The Flower box is not empty:\n");

return 0;

}

else

{

top++;

printf("Enter the flower id that you want to putin:\n");

scanf("%d", &stack[top]);

printf("your %d is succssfully Addede\n", stack[top]);

}

break;

case 2:

if (top == -1)

{

printf("There is Flower to remove\n");

}

else

{

top--;

printf("Successfully Flower to remove %d:\n", stack[top]);

}

break;

case 3:

if (top == -1)

{

printf("No Flower found\n");

return 0;

}

else

{

printf("Your Flowers is:\n");

for (int i = 0; i <= top; i++)

{

printf("%d\n", stack[i]);

}

}

break;

default:

return 0;

break;

}

}

}

int main()

{

int choice;

int in;

printf("\nWelcome to Data Structure Lab Project\n\n");

printf("\n\nData Structure's Applications\n\n");

printf("\n MENU \n");

printf("\n 1.Login \n");

printf("\n 2.Registration \n");

printf("Enter your choice:\t");

scanf("%d", &in);

switch (in)

{

case 1:

login();

break;

case 2:

printf("Enter user Name:\n");

scanf("%d", &userName[counter]);

printf("Enter user Password:\n");

scanf("%d", &password[counter]);

counter++;

printf("Enter for login:\n");

login();

break;

default:

printf("\n Wrong Choice:\n");

break;

}

while (1)

{

printf("\n\t\t MENU \n\n");

printf("\n 1.Sorting application: \n");

printf("\n 2.Linkedlist application: \n");

printf("\n 3.Stack application: \n");

printf("\n 4.Queue application: \n");

printf("\n 5.Exit \n\n\n");

printf("Enter your choice:\t");

scanf("%d", &choice);

switch (choice)

{

case 1:

Sorting();

break;

case 2:

Linkedlist();

break;

case 3:

Stack();

break;

case 4:

Qeue();

break;

case 5:

exit(0);

break;

default:

printf("\n Wrong Choice:\n");

break;

}

}

return 0;

}

int Linkedlist()

{

int choice;

while (1)

{

printf("\n Book-self Record Management System\n\n \n");

printf("\n 1.Create a self \n");

printf("\n 2.Display Th book \n");

printf("\n 3.Push at the beginning \n");

printf("\n 4.Push at the end \n");

printf("\n 5.Push at specified position \n");

printf("\n 6.Remove from beginning \n");

printf("\n 7.Remove from the end \n");

printf("\n 8.Remove from specified position \n");

printf("\n 9.Exit \n");

printf("Enter your choice:\t");

scanf("%d", &choice);

switch (choice)

{

case 1:

create\_linkedList();

break;

case 2:

display\_linkedList();

break;

case 3:

insertBegin\_linkedList();

break;

case 4:

insertEnd\_linkedList();

break;

case 5:

insertAnyPosition\_linkedList();

break;

case 6:

deleteAtBeginning\_linkedList();

break;

case 7:

deleteAtEnd\_linkedList();

break;

case 8:

deleteAtPosition\_linkedList();

break;

case 9:

exit(0);

break;

default:

printf("\n Wrong Choice:\n");

main();

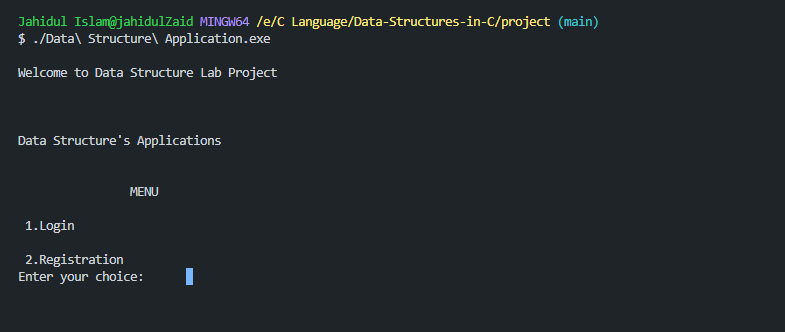
break;

}

}

}

## Output Of the Initial Page



# Chapter 3 Performance Evaluation

## Simulation Environment/ Simulation Procedure

The system maintains records of Every Data Structure algorithm , the courses and modules on which they are registered and the outcome of their studies. The application is distributed application because the data is stored centrally by the registration database administer that accessed by academic staff and administrators across the tvet’s employees. Students also access the system to view their results. The real system would be service oriented and would expose some of its functionality as a set of services which could be consumed by other application.

## 

# Chapter 4 Conclusion

## 4.1 Scope of Future Work

It can be applied to a larger institutes. We also can develop a gui application.

# References

1. Lab report