A picture containing text

Description automatically generated



Lab Journal: X

Date:

Student : Lotfullah Muslimwal

Enrollment : 01-131232-039

Data Structre And Algorithms (Fall-2024)

Teacher: Engr. Aleem Ahmad

Department of Software Engineering

Bahria University, Islamabad

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Task No:** | **Task Wise Marks** | | **Documentation Marks** | | **Total Marks**  **(20)** |
| **Assigned** | **Obtained** | **Assigned** | **Obtained** |
| 1 | 3 |  | 5 |  |  |
| 2 | 3 |  |
| 3 | 3 |  |
| 4 | 3 |  |
| 5 | 3 |  |

**Comments:**

**Signature**

# Lab No: 05

**Task 1: Priority Queue Implementation with Role-based Classification**

**Code:**#include<iostream>

// this two libraries is used for vilidation

#include<limits>

#include<sstream>

using namespace std;

int arr\_size; // store the total size of the queue which is set by the user.

template<class T>

class Queue {

private:

int front, rear, count; // total element in the queue

T\* array; // declar array to store element in queue

public:

Queue() {

front = rear = -1;

array = new T[arr\_size]; // uered chosed size

count = 0;

}

bool isEmpty() {

return (front == -1 && rear == -1);

}

bool isFull() {

return (rear == arr\_size - 1);

}

//passing the type and value

void enQueue(T value) {

if (!isFull()) {

if (front == -1) {

front = 0;

}

rear++;

array[rear] = value;

count++;

}

else {

cout << "Queue Overflow!" << endl;

}

}

// we use T because its class template so it should be that type

T deQueue() {

T val; // we use this for storing it temporary

if (!isEmpty()) {

val = array[front]; //Store the front element in a temporary variable

front++;

return val; //returned the removed element

}

else {

cout << "Queue Underflow!" << endl;

}

return ""; // return empty string

}

int getSize() {

return count; // return total element size

}

T peek() {

if (!isEmpty()) {

return array[front];

} else {

cout << "Queue is empty!" << endl;

return "";

}

}

void displayQueue() {

if (isEmpty()) {

cout << "Queue is empty!" << endl;

return;

}

cout << "Queue elements: ";

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

cout << array[i] << " ";

}

cout << endl;

}

};

void displayMenu() {

cout << "\n\t\t\t\*\*\* Priority Queue Menu \*\*\*" << endl;

cout << "\t1. Enqueue a person (Administrator, Faculty, Student)" << endl;

cout << "\t2. Dequeue the highest priority person" << endl;

cout << "\t3. Peek at the highest priority person" << endl;

cout << "\t4. Display the size of the queue" << endl;

cout << "\t5. Display entire queue by priority" << endl;

cout << "\t6. Exit" << endl;

cout << "\n\tEnter your choice: ";

}

int getValidatedInput() {

string input;

int result;

while (true) {

getline(cin, input);

if (input.empty() || input.find\_first\_not\_of(" \t\n\r") == string::npos) {

cout << "\n\tEmpty input. Please enter a valid number: ";

continue;

}

//it convert strings to inputA

stringstream ss(input);

if (ss >> result) {

return result; // return an valid int

}

else {

cout << "\n\tInvalid input. Please enter a valid number: ";

}

}

}

string getValidatedString() {

string input;

while (true) {

getline(cin, input);

if (input.empty() || input.find\_first\_not\_of(" \t\n\r") == string::npos) {

cout << "\n\tEmpty input. Please enter a valid name: ";

continue;

}

return input;

}

}

int getValidatedRole() {

int role;

while (true) {

cout << "\n\tChoose the role (1. Administrator, 2. Faculty, 3. Student): ";

role = getValidatedInput();

if (role == 1 || role == 2 || role == 3) {

return role;

} else {

cout << "\n\tInvalid choice. Please enter 1, 2, or 3.";

}

}

}

int main() {

int choice;

string value;

// Input validation for size

cout << "\n\t\t\t\*\*\* Welcome to the Priority Queue Program \*\*\*" << endl;

cout << "\n\tEnter the size of the queue: ";

arr\_size = getValidatedInput();

//defrent queues for each user

Queue<string> Admin;

Queue<string> Faculty;

Queue<string> Student;

Queue<string> priorityQueue;

do {

displayMenu();

choice = getValidatedInput();

switch (choice) {

case 1: {

cout << "\n\tEnter the name of the person: ";

string name = getValidatedString();

int roleChoice = getValidatedRole();

// after chosing the person we chose the person job

if (roleChoice == 1) {

Admin.enQueue(name);

cout << "\t\*\*\* " << name << " (Administrator) added successfully \*\*\*" << endl;

} else if (roleChoice == 2) {

Faculty.enQueue(name);

cout << "\t\*\*\* " << name << " (Faculty) added successfully \*\*\*" << endl;

} else if (roleChoice == 3) {

Student.enQueue(name);

cout << "\t\*\*\* " << name << " (Student) added successfully \*\*\*" << endl;

}

break;

}

case 2: {

// dequeu by sequence

if (!Admin.isEmpty()) {

cout << "\t\*\*\* " << Admin.deQueue() << " (Administrator) dequeued successfully \*\*\*" << endl;

} else if (!Faculty.isEmpty()) {

cout << "\t\*\*\* " << Faculty.deQueue() << " (Faculty) dequeued successfully \*\*\*" << endl;

} else if (!Student.isEmpty()) {

cout << "\t\*\*\* " << Student.deQueue() << " (Student) dequeued successfully \*\*\*" << endl;

} else {

cout << "\t\*\*\* Queue is empty \*\*\*" << endl;

}

break;

}

case 3: {

// printing the front person

if (!Admin.isEmpty()) {

cout << "\t\*\*\* Front Person: " << Admin.peek() << " (Administrator) \*\*\*" << endl;

} else if (!Faculty.isEmpty()) {

cout << "\t\*\*\* Front Person: " << Faculty.peek() << " (Faculty) \*\*\*" << endl;

} else if (!Student.isEmpty()) {

cout << "\t\*\*\* Front Person: " << Student.peek() << " (Student) \*\*\*" << endl;

} else {

cout << "\t\*\*\* Queue is empty, nothing to peek \*\*\*" << endl;

}

break;

}

case 4: {

//printing the fromt person

int totalSize = Admin.getSize() + Faculty.getSize() + Student.getSize();

cout << "\t\*\*\* Current size of the queue: " << totalSize << " \*\*\*" << endl;

break;

}

case 5: {

cout << "\n\t\*\*\* Displaying the Queue by Priority \*\*\*" << endl;

Admin.displayQueue();

Faculty.displayQueue();

Student.displayQueue();

break;

}

case 6:

cout << "\n\t\*\*\* Exiting the program... \*\*\*" << endl;

break;

default:

cout << "\n\tInvalid choice, please try again." << endl;

}

} while (choice != 6);

return 0;

}

**GitHub-Link:** [**https://github.com/lotfullahmsl/DSA-Lab-FA2024**](https://github.com/lotfullahmsl/DSA-Lab-FA2024)

**Screenshot:**

