Drone Delivery Management System

ELEMENTARY DATA SYSTEMS AND LOGICAL THINKING

Dhanyashri V - ME24B1050

Question 6:

Designing a cargo drone traffic controller to manage drone deliveries in a busy airspace. The system is designed to manage the logistical operations of a fleet of drones. The core functionalities include:

- **Request and dispatch:** Customers can place delivery requests, specifying the type of goods to be delivered (food, medicine, tools, etc.), among 6 delivery requests.
  - This is done with the help of queues.
- **Priority Dispatching:** The system dispatches drones to fulfill delivery requests, with an option for priority dispatch.
  - This is done with the help of stacks
- **Flight Log:** The system maintains a log of recent drone dispatches, recording the type of delivery and if it is full, the oldest log is archived.
  - Done with the help of arrays.
- **Maintenance Tracker:** The system tracks the status of drones, including overloaded drones(singly linked list) , drones in service(doubly linked list) , and drones requiring emergency rerouting(circular linked list).
  - Help of linked lists.

DESIGN OF THE CODE

The C code implements the following components:

- **Data Structures:**
    - **Queue (Circular Array):** The inp_arr array is used as a circular queue to manage incoming delivery requests. New requests go to the back and the system acquires them from the front. *Acts like a waiting line.*
    - **Flight Log (Circular Array):** The flight_log array stores a history of the most recent dispatches.
    - **Singly Linked List:** The node structure is used to maintain a list of overloaded drones - since they carry too much, new info should be quickly shifted to another drone, which is allowed by creating a new singly linked list since it cannot take up more space.
    - **Doubly Linked List:** The dnode structure is used to maintain a list of drones that are in service - these drones are currently making deliveries and have to choose back and forth between options.
    - **Circular Singly Linked List:** The cnode structure is used to manage drones that require emergency rerouting - it can traverse to any node which helps during emergencies.

- **Functions:**
    - logval(): Adds a delivery log entry to the flight_log array.
    - array(): Prints the delivery type based on the request code.
    - enqueue(): Adds a delivery request to the inp_arr queue.
    - dequeue(): Processes a standard delivery request from the queue.
    - pop(): Processes a priority delivery request (from the top of the queue).
    - createNodeList(): Creates a singly linked list of overloaded drones.

- insert(): Inserts a drone into the doubly linked list of serviced drones.
- addnode(): Adds a drone to the circular linked list for emergency rerouting.
- deletenode(): Removes a drone from the emergency rerouting list.
- main(): The main function provides a menu-driven interface for the system.

Output interface with user

1. The main() function presents a menu of options to the user.
   The user can choose to:
   - Add a delivery request (option 1).
   - Dispatch a drone (option 2).
   - View the flight log (option 3).
   - View/manage drone statuses (option 4).
   - Exit the program (option 5).
2. Delivery requests are added to the inp_arr queue using the enqueue() function.
3. Drones are dispatched using either the dequeue() function (for normal dispatch) or the pop() function (for priority dispatch).
4. The logval() function updates the flight_log with the details of each dispatch.
5. The system uses linked lists to manage drone statuses: overloaded, in service, and emergency rerouting.

CODE

```
#include <stdio.h>
#include <stdlib.h>
#define SIZE 10

//creating array for input requests and storing flight log
int inp_arr[10],flight_log[6];
int Rear = - 1, Front = -1, top = -1, val = 0, value, lv;
//using circular array for last 6 dispatches
void logval(int value)
{
        if(val!=6)
        {
        flight_log[val]=value;
        val=val+1;
        }
        else
        {
        val=0;
        logval(value);
        }
}
//to display the output as per desired request
void array(int num)
{
        if(num==1)
        {
        printf("food\n");
        }
        else if(num==2)
        {
    printf("medicine\n");
        }
        else if(num==3)
        {
        printf("tools\n");
        }
        else if(num==4)
        {
        printf("water\n");
        }
        else if(num==5)
        {
        printf("parts\n");
        }
        else if(num==6)
        {
        printf("fuel\n");
```

```
        }
}
//to add an element(delivery request) to the queue
void enqueue()
{
        int insert_item;
        if (Rear == SIZE - 1)
        printf("requests overflow ; wait for the request to be done \n");
        else
        {
        if (Front == - 1)

        Front = 0;
        scanf("%d", &insert_item);
        printf("request accepted \n");
        Rear = Rear + 1;
        top=top+1;
        inp_arr[Rear] = insert_item;
        }
}
//to remove an element- here stands for dispatching the requests
void dequeue()
{
        if (Front == - 1 || Front > Rear)
        {
        printf("no requests in the queue \n");
        return ;
        }
        else
        {
        printf("request in progress is :\t");
        array(inp_arr[Front]);
        lv=lv+1;
        logval(inp_arr[Front]);
        value=inp_arr[Front];
        Front = Front + 1;
        }
}
//removes the top element- helps with priority dispatch which is the first request in the queue
void pop()
{
        if (top == -1)
        {
        printf("no request in the queue");
        }
        else
        {
        printf("request in progress is \t");
        array(inp_arr[top]);
        logval(inp_arr[top]);
```

```
        lv=lv+1;
        top = top - 1;
        Rear=Rear-1;

        }
}
//forming a linked list for overloaded drones
struct node
{
        char data[20];
        struct node *next;
} ;

struct node *head,*newnode,*temp;
//adding nodes to the overloaded drone
void createNodeList()
{
        newnode=(struct node *)malloc(sizeof(struct node));
        printf("enter the drone name to enter:\n");
        scanf("%s",&newnode->data);
        newnode->next=0;
        if(head==NULL)
        {
        head=newnode;
        temp=newnode;
        }
        else
        {
        temp->next=newnode;
        temp=newnode;
        }

}
//double liked list for drones with ready usage
struct dnode
{
        char data[20];
        struct dnode *next;
        struct dnode *prev;
};

struct dnode *dhead,*dnewnode,*dtemp,*duse,*trav;

//adding names to the doubly list
void insert()
{
        dnewnode=(struct dnode*)malloc(sizeof(struct dnode));
        printf("enter the name of the drone to include\n");
   scanf("%s",&dnewnode->data);
        dnewnode->next = 0;
```

```
        if (dhead == NULL)
        {
        dhead = dnewnode;
        dtemp=dnewnode;
        dnewnode->prev = 0;
        duse=dhead;
        }
        else
        {
        dtemp->next=dnewnode;
        dnewnode->prev=dtemp;
        dnewnode->next=0;
        dtemp=dnewnode;
        }
}
//circular linked list for emergency rerouting – fills in the same list
struct cnode
{
        char data[20];
        struct cnode * next;
};

struct cnode *chead,*cnewnode,*ctemp;

//applying emergency rerouting to the drones
void addnode()
{
        cnewnode=(struct cnode *)malloc(sizeof(struct cnode));
        printf("enter the name of emergency drone to be rerouted:\n");
   scanf("%s",&cnewnode->data);
        if(chead==NULL)
        {
        chead=cnewnode;
        ctemp=cnewnode;
        cnewnode->next=chead;
        }
        else
        {
        cnewnode->next=chead;
        ctemp->next=cnewnode;
        ctemp=cnewnode;

        }
}
//releasing drones (deleting nodes) when the weather clears
void deletenode()
{
        printf("the emergency drone that is rerouted is : %s",chead->data);
        chead=chead->next;
        ctemp->next=chead;
```

```
}
//driving function
int main()
{
	int opt,option,choice,any;
	while(opt!=5)
	{
	printf("enter the service to be provided\n");
	printf("1.add delivery request\n2.dispatch\n3.flight log unit\n4.maintenance tracker\n5.exit\n");
	scanf("%d",&opt);
	switch(opt)
	{
	case 1:   // case for delivery request

	printf("enter the delivery request required");
	printf("1. food\n2.medicine\n3.tools\n4.water\n5.parts\n6.fuel\n");
	enqueue();  // function call to input values
	break;


	case 2:   //case for dispatching

	printf("is it a priority dispatch or normal way dispatch??");
	printf("if priority press 1 or press any other number:");
	scanf("%d",&option);
	if(option==1)  // if it a priority dispatch , dispatching the recently added delivery request
	{
	pop();
	if(duse->next=0)  // assigning the drones for the delivery request
	{
		printf("no drones available to dispatch,add drones in serviced drone list\n");
	}
	else
	{
		printf("assigning the dispatch to drone named : %s\n",&duse->data);
		duse=duse->next;
	}
	}

	else   // if it is a normal dispatch, dequeue the needed dispatch
	{
	dequeue();
	if(duse->next=0)  // assigning the drones for the dispatch
	{
		printf("no drones available to dispatch,add drones in serviced drone list\n");
	}
	else
	{
		printf("assigning the dispatch to drone named : %s\n",&duse->data);
		duse=duse->next;
```

```
}

}
break;

case 3:   // displying the flight log

printf("Displaying the last %d dispatches done\n ",lv);
for(int i=0;i<lv;i++)
{
array(flight_log[i]);
}
break;

case 4:  // to track the overloaded drones , drones in usage , drones for rerouting and knowing the latest dispatches and the assigned drones

printf("maintenance tracker\n");

printf("enter a choice\n1.overloaded drones\n2.seviced drones\n3.emergency drones\n4.get list of recently dispatched drones");
int select;
scanf("%d",&choice);

switch(choice)
{
case 1: // singly list for the drones in overload condition

printf("do u want to 1.display names or 2. input names\n");
  scanf("%d",&select);
if(select==1)
{
        temp=head;
        while(temp!=0)
        {
      printf("%s\n",temp->data);
        temp=temp->next;
        }
}
else if(select==2)
{
        createNodeList();
}

else
{
        printf("not a valid option");
}

break;

case 2: // doubly list for drones in usage
```

```
printf("do u want to 1.display names or 2. input names\n");
  scanf("%d",&select);
if(select==1)
{
            dtemp=dhead;
            while(temp!=0)
            {
        printf("%s\n",dtemp->data);
            dtemp=dtemp->next;
            }
}
else if(select==2)
{
            insert();
}

else
{
            printf("not a valid option");
}

break;

case 3:  // circular list for emergency rerouting drones

printf("select one option\n1.add drone\n2.release drone\n3.display drones");
scanf("%d",&any);
switch(any)
{
            case 1:
            addnode();
            break;

            case 2:
            deletenode();
            break;

            case 3:
            ctemp=chead;
            do
            {
         printf("%s\n",&ctemp->data);
            ctemp=ctemp->next;
            }
            while(ctemp!=chead);

            break;

            default:
```

```
                printf("enter a valid option!!! exiting options!!\n");
                break;
        }
        break;

        case 4:  // displying the recent dispatches

        if(duse->prev==dhead)
        {
                printf("no recent dispatches done \n");
        }
        else
        {
                printf("recently dispatched drones and delivery\n");
                int a =0;
                trav=duse->prev;
                while(trav->prev!=dhead)
                {
                printf("%s %s",trav->data,flight_log[a]);
                trav=trav->prev;
                a=a+1;
                }

        }


        default :
        printf("not an valid option !!!exiting!!!\n");
        }
        break;

        case 5 :
        printf("program exiting\n");
        break;

        default:
        printf("not a valid choice!!!\n");
        break;
        }
        }
}
```

OUTPUT

```
enter the service to be provided
1.add delivery request
2.dispatch
3.flight log unit
4.maintanence tracker
5.exit
1
enter the delivery request required1. food
2.medicine
3.tools
4.water
5.parts
6.fuel
5
request accepted
enter the service to be provided
1.add delivery request
2.dispatch
3.flight log unit
4.maintanence tracker
5.exit
2
is it a priority dispatch or normal way dispatch??if priority press 1 or press any other
    number:1
request in progress is  parts
```