بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ

3- Queues: ************************************

Using arrays

```c
#include <stdio.h>
#include <stdlib.h>

#define SIZE 100
int queue[SIZE];
int rear = -1, front = -1;


void enQueue(int value)
{
      if (rear == SIZE - 1)
            printf("Overflow, queue is full\n");
      else
      {
            if (front == -1)
                  front = 0;
            rear++;
            queue[rear] = value;
            printf("Value %d added to the queue successful.\n", value);
      }
}


void deQueue()
{
      if (rear == front)
            printf("Underflow, queue is already empty!\n");
      else
      {
            printf("%d deleted\n", queue[front]);
            front++;
            if (front == rear)
                  rear = front = -1;
      }
}


void display_array()
{
      printf("\n------------------------------------------------------------\n");
      if (rear == -1)
            printf("Queue is empty!\n");
      else
      {
            for (int i = front; i <= rear; i++)
                  printf(" %d |", queue[i]);
      }
      printf("\n------------------------------------------------------------\n");
}
```

```c
int main()
{
     while (1)
     {
          int value, choice;
          printf("TYPE 01 ---> push\n");
          printf("TYPE 02 ---> pop\n");
          printf("\nchoice = ");
          scanf("%d", &choice);
          switch (choice)
          {
          case 1:
               printf("\nValue = ");
               scanf("%d", &value);
               enQueue(value);
               display_array();
               break;
          case 2:
               deQueue();
               display_array();
               break;

          default:
               printf("no valid\n");
               break;
          }
     }
     return 0;
}
```

Using linked list

```c
#include <stdio.h>
#include <stdlib.h>

typedef struct Queue
{
     int data;
     struct Queue *next;
} queue;

queue *head = NULL;

queue *create_node(int value)
{
     queue *node = (queue *)malloc(sizeof(queue));
     node->data = value;
     node->next = NULL;
     return node;
}

void enQueue(int value)
{
     queue *element = create_node(value);
     if (head == NULL)
          head = element;
     else
     {
          queue *temp = head;
          while (temp->next != NULL)
               temp = temp->next;
          temp->next = element;
     }
}

void deQueue()
{
     if (head == NULL)
          printf("Underflow, queue is empty!\n");
     else
     {
          queue *temp = head;
          if (temp->next == NULL)
          {
               head = NULL;
               free(temp);
          }
          else
          {
               head = temp->next;
               free(temp);
          }
     }
}
```

```c
void display()
{
	printf("\n-----------------------------------------------------\n");
	if (head == NULL)
		printf("List is empty!\n");
	else
	{
		queue *temp = head;
		while (temp->next != NULL)
		{
			printf(" %d | ", temp->data);
			temp = temp->next;
		}
		printf(" %d | ", temp->data);
	}
	printf("\n-----------------------------------------------------\n");
}

int main()
{
	while (1)
	{
		int value, choice;
		printf("TYPE 01 ---> push\n");
		printf("TYPE 02 ---> pop\n");
		printf("\nchoice = ");
		scanf("%d", &choice);
		switch (choice)
		{
		case 1:
			printf("\nValue = ");
			scanf("%d", &value);
			enQueue(value);
			display();
			break;
		case 2:
			deQueue();
			display();
			break;

		default:
			printf("no valid\n");
			break;
		}
	}
	return 0;
}
```

Created by: Mohammed El Idrissi Laoukili

لاإله الا الله

Instagram : __elidrissii