UCS 1312: Data Structures Lab

**Aditya Krishna P**

**185001010**

**CSE-A**

**Exercise 6: Application of queue**

**·** Create queue ADT as a header file “queue.h”.

· Create node with two fields namely job number and burst time (jno, bt).

· Assume there are two queues Q1 and Q2.

· Insert the data given below in Q1 and Q2 based on minimum waiting time.

(J1, 6), (J2, 5), (J3, 2), (J4, 3), (J5, 7), (J6, 3), (J7, 7), (J8, 2), (J9, 3) and (J10, 7).

· Compute the average waiting time of Q1 and Q2

· Display both the queues with average waiting time.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**queue.h**

typedef struct \_node

{

int bt, wt;

char jno[5];

struct \_node \* next;

} node;

node \*front1 =NULL, \*rear1= NULL, \*front2=NULL, \*rear2=NULL;

void enqueue (char j[], int b, int x)

{

node \* new= (node \*) malloc (sizeof(node));

strcpy(new->jno, j);

new->bt=b;

new->next=NULL;

if(x==1) {

if(front1 == NULL)

{ new->wt=0;

front1=rear1=new; }

else

{ new->wt= rear1->wt + rear1->bt;

rear1->next= new;

rear1=new; } }

else if(x==2) {

if(front2 == NULL)

{ new->wt=0;

front2=rear2=new; }

else

{ new->wt= rear2->wt + rear2->bt;

rear2->next= new;

rear2=new; } } }

void dequeue (int x)

{ if (x==1) {

if(front1 ==NULL)

printf ("Empty Queue \n");

else

{ node\* temp=front1;

front1=front1->next;

printf("Deleted %s %d\n", temp->jno, temp->bt);

free(temp); } }

else if (x==2) {

if(front2 ==NULL)

printf ("Empty Queue \n");

else

{ node\* temp=front2;

front2=front2->next;

printf("Deleted %s %d\n", temp->jno, temp->bt);

free(temp); } } }

void display(int x)

{

if(x==1) {

printf("---QUEUE 1---\n");

if (front1==NULL)

printf("Empty Queue\n");

else

{ printf("JOB NO.\tBURST TIME\n");

node \* temp= front1;

while(temp!=NULL)

{

printf("%s\t %d\n", temp->jno, temp->bt);

temp=temp->next;

} } }

else if(x==2) {

printf("---QUEUE 2---\n");

if (front2==NULL)

printf("Empty Queue\n");

else

{ printf("JOB NO.\tBURST TIME\n");

node \* temp= front2;

while(temp!=NULL)

{

printf("%s\t %d\n", temp->jno, temp->bt);

temp=temp->next;

} } } }

void insert (char j[], int b)

{

int t1= rear1->bt+ rear1->wt;

int t2= rear2->bt+ rear2->wt;

if(t1<=t2)

enqueue(j,b,1);

else

enqueue(j,b,2); }

float avgwt(int x)

{

float total=0.0;

int ct=0;

if(x==1)

{

if (front1==NULL)

printf("Empty Queue\n");

else

{ node \* temp= front1;

while(temp!=NULL)

{ total+=temp->wt;

ct++;

temp=temp->next;

} } }

else if(x==2)

{

if (front2==NULL)

printf("Empty Queue\n");

else

{ node \* temp= front2;

while(temp!=NULL)

{ total+=temp->wt;

ct++;

temp=temp->next;

} } }

return (total/ct);}

**q.c**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include "queue.h"

int main()

{

printf("Inserting the following data based on minimum waiting time\n");

printf("(J1, 6), (J2, 5), (J3, 2), (J4, 3), (J5, 7), (J6, 3), (J7, 7), (J8, 2), (J9, 3) and (J10, 7).\n\n");

enqueue("J1", 6, 1);

enqueue("J2", 5, 2);

insert("J3", 2);

insert("J4", 3);

insert("J5", 7);

insert("J6", 3);

insert("J7", 7);

insert("J8", 2);

insert("J9", 3);

insert("J10", 7);

display(1);

printf("AVG WAITING TIME = %f\n", avgwt(1));

printf("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n");

display(2);

printf("AVG WAITING TIME = %f\n", avgwt(2));

printf("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n");

return 0;

}

**OUTPUT**

C:\Users\Aditya\Desktop ./q.out

Inserting the following data based on minimum waiting time

(J1, 6), (J2, 5), (J3, 2), (J4, 3), (J5, 7), (J6, 3), (J7, 7), (J8, 2), (J9, 3) and (J10, 7).

---QUEUE 1---

JOB NO. BURST TIME

J1 6

J4 3

J6 3

J7 7

J10 7

AVG WAITING TIME = 9.200000

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

---QUEUE 2---

JOB NO. BURST TIME

J2 5

J3 2

J5 7

J8 2

J9 3

AVG WAITING TIME = 8.400000

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_