#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#define SIZE 10

struct q

{

char vnum[SIZE][12];

char vtype[SIZE][SIZE];

int cost[SIZE];

int front;

int rear;

};

typedef struct q QUEUE;

QUEUE create\_queue()

{

QUEUE q1;

q1.front = -1;

q1.rear = 0;

return q1;

}

void enqueue(QUEUE \*q1)

{

char vcnum[10], vctype[5];

if(q1->rear == SIZE)

printf("Queue Overflow\n");

else

{

printf("Enter vehicle number, vehicle type (as LMV or HMV)\n");

scanf("%s%s", vcnum, vctype);

strcpy(q1->vnum[q1->rear],vcnum);

strcpy(q1->vtype[q1->rear], vctype);

if(strcmp(q1->vtype[q1->rear],"HMV")==0)

q1->cost[q1->rear]= 100;

else

q1->cost[q1->rear]= 50;

q1->rear ++;

}

}

void dequeue(QUEUE \*q1)

{

if(q1->rear == q1->front +1)

printf("Queue Underflow\n");

else

{

q1->front++;

printf("Vehicle moving out of toll: %s. %s. \n", q1->vnum[q1->front], q1->vtype);

}

}

void displayqueue(QUEUE \*q1)

{

int i;

if(q1->rear == q1->front+1)

{

printf("Queue is empty\n");

}

else

{

printf("Vehicles at Toll bay are: \n");

for(i=q1->front+1; i<q1->rear; i++)

{

printf("%s %s %d\n", q1->vnum[i], q1->vtype[i], q1->cost[i]);

}

printf("\n");

}

}

void predicthmvcost(QUEUE \*q1)

{

int i, totalhmvcost=0;

if(q1->rear == q1->front+1)

{

printf("Queue is empty\n");

}

else

{

for(i=q1->front+1; i<q1->rear; i++)

{

if(strcmp(q1->vtype[i], "HMV")==0)

totalhmvcost = totalhmvcost + q1->cost[i];

}

printf("Predicted Total cost of HMV is %d\n", totalhmvcost);

}

}

void predictlmvcost(QUEUE \*q1)

{

int i, totallmvcost=0;

if(q1->rear == q1->front+1)

{

printf("Queue is empty\n");

}

else

{

for(i=q1->front+1; i<q1->rear; i++)

{

if(strcmp(q1->vtype[i], "LMV")==0)

totallmvcost = totallmvcost + q1->cost[i];

}

printf("Predicted Total cost of LMV is %d\n", totallmvcost);

}

}

void displayqueuevehicles(QUEUE \*q1)

{

int i, lmvcount=0;

if(q1->rear == q1->front+1)

{

printf("Queue is empty\n");

}

else

{

printf("Vehicles at Toll bay are: \n");

for(i=q1->front+1; i<q1->rear; i++)

{

if(strcmp(q1->vtype[i], "HMV")==0)

printf("%s %s %d\n", q1->vnum[i], q1->vtype[i], q1->cost[i]);

else

lmvcount++;

}

printf("\n");

}

printf("Number of LMV vehicles waiting to pass toll are: %d\n", lmvcount);

}

int main()

{

int i, n;

QUEUE q11, \*q1;

q11 = create\_queue();

q1=&q11;

printf("Enter number of vehicles waiting at Toll bay?\n");

scanf("%d",&n);

if(n>SIZE)

{

printf("Number of vehicles to enqueue are more than the size of toll bay.\n");

}

else

{

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

{

enqueue(q1);

}

displayqueue(q1);

printf("HMV vehicles are: \n");

displayqueuevehicles(q1);

predicthmvcost(q1);

predictlmvcost(q1);

}

return 0;

}

2 tirumala

#include <stdlib.h>

#include<stdio.h>

typedef struct node

{

int data;

struct node \*next;

}\*List;

int count=0;

List create();

List insertrear(List,int);

void display(List);

List insertatpos(List first,int item,int pos);

int main()

{

List L=NULL;

int ele,ch,m,i;

printf("Enter number of pilgrims\n");

scanf("%d",&ch);

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

{

printf("enter age of pilgrim\n");

scanf("%d",&ele);

if(count==0)

L=insertrear(L,ele);

else

{

if(L->data<ele)

{

m=count/2;

L=insertatpos(L,ele,m);

}

else

L=insertrear(L,ele);

}

}

display(L);

return 0;

}

List create()

{

List x;

x=malloc(sizeof(struct node));

count++;

x->next=NULL;

return x;

}

List insertrear(List L,int x)

{

List temp,p=L;

temp=create();

temp->data=x;

if(L==NULL)

{

L=temp;

}

else{

while(p->next!=NULL)

{

p=p->next;

}

p->next=temp;

}

return L;

}

void display(List L)

{

if(L==NULL)

{

printf("List is empty\n");

}

else

{

while(L!=NULL)

{

printf("%d ",L->data);

L=L->next;

}

}

}

List insertatpos(List first,int item,int pos)

{

List t,prev,cur;

int c=0;

t=create( );

t->data=item;

cur=first;

prev=NULL;

if(pos==0)

{

first->next=t;

//return first;

}

else{

while(cur->next!=NULL && c!=pos)

{

c++;

prev=cur;

cur=cur->next;

}

prev->next=t;

t->next=cur;

}

return first;

}