**Problem 1:** Implement the priority queue scheduling algorithm using linked list.

**Answer:**

Source code:

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

struct Node

{

char process[3];

int priority;

struct Node\* next;

};

struct Node\* Head= NULL;

// print all elements of queue

void print(){

printf("printing queue...\t");

struct Node \*temp = Head;

if(temp->next==NULL){

printf("|%s|%d|",temp->process,temp->priority);

}else{

while(temp != NULL){

printf("|%s|%d|",temp->process,temp->priority);

if(temp->next != NULL){

printf("-->");

}

temp=temp->next;

}

}

printf("\n");

}

// delete all nodes before exiting

void ext(){

struct Node \*trav = Head;

struct Node \*temp = Head;

while(trav!=NULL){

temp=trav;

trav=trav->next;

free(temp);

}

printf("Traversed\n");

}

// add element to queue

void enqeue(int priority, char\* process){

printf("Adding...\n");

struct Node\* temp = (struct Node\*)malloc(sizeof(struct Node));

temp->priority=priority;

strcpy(temp->process,process);

temp->next=NULL;

if(Head==NULL){

Head=temp;

}else{

if(priority < Head->priority){

temp->next=Head;

Head = temp;

}else{

// traverse till dont find the correct priority

// then insert node at the best position

struct Node\* trav= Head;

while(trav->next!=NULL){

if(trav->next->priority > priority){

break;

}

trav=trav->next;

}

temp->next=trav->next;

trav->next=temp;

}

}

print();

}

// executing process

void execute(){

struct Node \*temp = Head;

if(temp==NULL){

printf("No processes left.\n");

}else if(temp->next==NULL){

free(temp);

printf("All processes finished.\n");

}else{

Head=temp->next;

printf("%s process having priority %d executed.\n",temp->process,temp->priority);

free(temp);

print();

}

}

void main()

{

int choice = 0, br=0;

while (choice!=3 && br!=1)

{

choice=0;

printf("To ADD PROCESS press 1 \nTo EXECUTE PROCESS press 2 \nTo EXIT press 3\n");

scanf("%d",&choice);

switch (choice)

{

case 1:;

// add node according to priority

char \*n;

n=(char\*)malloc(sizeof(char)\*3);

int i=0;

printf("Enter process name\n");

scanf("%s",n);

printf("Enter PRIORITY\n");

scanf("%d",&i);

enqeue(i,n);

break;

case 2:

// execute process

execute();

break;

case 3:

// Exit the process and clear space

ext();

printf("EXITING...\n");

break;

default:

//print error and breaking loop

printf("ILLEGAL INPUT\n");

br=1;

break;

}

}

}

Screen shots of code output:



