DSA ASSIGNMENT 7

TATHAGATA GHOSH
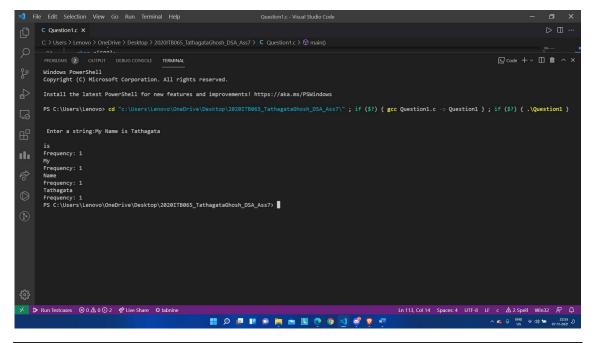
INFORMATION TECHNOLOGY

2020ITB065

HY

```c
//Tathagata Ghosh - 2020ITB065 - HY - DSA Assignment 7

/*Q1. Write a program to categorize the words having the same length with
their frequency of
occurring in a given text. The list should display the frequency of each word
corresponding
to a specific length taken as an input. Use a linked list to implement.*/

#include<stdio.h>
#include<stdlib.h>
#include<string.h>

struct node
{
    char s[20];
    int strlength;
    int freq;
    struct node *next;
};

struct node *head;
struct node *newnode;

void insert(char s1[20])
{
    struct node *temp;
    struct node *t;
    t=(struct node *)malloc(sizeof(struct node));
    temp=(struct node *)malloc(sizeof(struct node));
    strcpy(temp->s,s1);
    temp->freq=1;
    temp->strlength=strlen(s1);
    int flag=0,mark=0;
    if(head==NULL)
    {
        head=temp;
        head->next=NULL;
    }
```

```c
        else
        {
            t=head;
            while(t!=NULL)
            {
                if(strcmp(s1,t->s)==0)
                {
                    flag=1;
                    t->freq ++;
                    break;
                }
                t=t->next;
            }
            if(flag==0)
            {
                if(head->strlength >= strlen(s1))
                {
                    temp->next=head;
                    head=temp;
                    mark=1;
                }
                else
                {
                    t=head;
                    while(t->next!=NULL)
                    {
                        if(t->next->strlength >= strlen(s1))
                        {
                            temp->next=t->next;
                            t->next=temp;
                            mark=1;
                            break;
                        }
                        t=t->next;
                    }
                }
                if(mark==0)
                t->next=temp;
            }
        }
}

void display()
{
    struct node *temp;
    temp=head;
    while(temp!=NULL)
    {
```

```c
        printf("\n");
        puts(temp->s);
        printf("Frequency: %d",temp->freq);
        temp=temp->next;
    }
}
int main()
{

    head=NULL;
    char c[500];
    printf("\n Enter a string:");
    gets(c);
    int j=0;
    while(c[j]!='\0')
    {
        int k=0;
        char s1[20];
        for(;c[j]!=' ' && c[j]!='\0';j++)
            {
                s1[k]=c[j];
                k++;
            }
        s1[k]='\0';
        insert(s1);
        if(c[j]=='\0')
            break;
        j++;
    }
    display();
}
```

OUTPUT:

//Tathagata Ghosh - 2020ITB065 - HY - DSA Assignment 7

/*Q2. Suppose 7 persons are recruited in a company to do their specific tasks. Each task requires
a fixed amount of time to complete and all those tasks are needed to be done in a single
platform that can be shared by each employee. But as per the regulations of the company
every employee should report to his/her immediate boss after doing the work every 2 units
of time only. The following table shows the name of the employee, their assigned task
number and total time required to complete their tasks. Find the order of the employees
with different time stamps to accomplish the assigned tasks. You have to show starting
time, ending time and the different time instants at which the employees shared that
common platform. Take name of the employee as input who will start work first from 0th
time instant. Use a linked list to implement.

Employee Name	Task Number	Time to Compute
A	2	10
B	5	11
C	7	5
D	1	8
E	3	19
F	6	23
G	4	11
*/

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

typedef struct Node
{
    char emp;
    int priority;
    int time;
    int start;
    int end;
    struct Node *next;
}node;

void insert(node **root, node el);
void display(node *root);
void roundrobin(node *root, char ch);

int ctr;

int main()
{
    int n;
    printf("Enter the number of employees\n");
    scanf("%d", &n);
    node *root = NULL;
//  node *formal = NULL;

    int i = 0;
    printf("Enter the data\n");

    for(i = 0; i<n; i++)
    {
        node el;
        el.emp = 'A'+i;
        printf("Employee %c \n", i+'A');
        el.start = el.end = -1;
        scanf("%d %d", &(el.priority), &(el.time));
        insert(&root, el);
    }

    printf("Enter the employee from which to start work\n");
    char ch;
    scanf(" %c", &ch);
//  printf("%c", ch);
    roundrobin(root, ch);
//  printf("The details of starting time and ending times are as follows\n");
    display(root);
```

```c
    return 0;
}

void roundrobin(node *root, char ch)
{
    int t = 0;
    node *cur = root;
    node *prev = root;
    while(cur->emp != ch)
    {
        prev = cur;
        cur = cur->next;
    }

    printf("%c %d %d\n", ch, 0, 2);

    cur->start = 0;
    t+=2;
    cur->end = t;
    cur->time -= 2;

//  cur = cur->next;
    if(cur->time <= 0)
    {
        //You got an IPhone
        prev->next = cur->next;
        node *mem = cur;
        free(mem);
        ctr--;
        cur = prev->next;
    }
//  printf("%u \n", cur);
//  printf("%u \n", cur->next);

    while(ctr > 0)
    {
        prev = cur;
//  printf("Next Emp is %c", cur->next->emp);

        cur = cur->next;
        if(cur->start == -1)
        {
            cur->start = t;
            cur->end = t;
        }
        if(cur->time > 2)
        {
            cur->time -= 2;
```

```c
            t += 2;
            cur->end = t;
            printf("%c %d %d\n", cur->emp, t-2, t);
        }
        else
        {
            if(cur->time > 0)
            {
                printf("%c %d %d \n", cur->emp, t, t+cur->time);
                t+= cur->time;
                cur->end = t;
            }
            prev->next = cur->next;
            node *mem = cur;
            free(mem);
            cur = prev->next;
            ctr--;
        }
    }
}

void insert(node **root, node el)
{
    node *val = (node *)malloc(sizeof(node));
    val->emp = el.emp;
    val->priority = el.priority;
    val->time = el.time;
    val->start = el.start;
    val->end = el.end;
    val->next = *root;

    ctr++;

    if(*root == NULL)
    {
        *root = val;
        val->next = *root;
        return;
    }
    if(ctr == 2)
    {
        (*root)->next = val;
        val->next = *root;
        return ;
    }
    if(el.priority <= (*root)->priority)
    {
        val->next = *root;
```

```c
        *root = val;
        node *cur = val;
        int num = ctr;
        while(num > 1)
        {
            cur = cur->next;
            num--;
        }
        cur->next = *root;
    }
    //case single node
    //case two nodes
    //case last node

    node *cur = *root;
    node *prev = *root;
    int count = ctr-1;
    while(count > 0 && (cur->priority <= el.priority))
    {
        prev = cur;
        cur = cur->next;
        count--;
    }

    prev->next = val;
    prev = prev->next;
    prev->next = cur;
}

void display(node *root)
{
    int num = ctr;
    node *cur = root;
    while(cur != NULL && ctr>0)
    {
        printf("%c %d %d %d\n", cur->emp, cur->start, cur->end, cur->time);
        cur = cur->next;
        ctr--;
    }
}
```

OUTPUT:

```
Enter the number of employees
5
Enter the data
Employee A
2 5
Employee B
1 8
Employee C
5 2
Employee D
3 9
Employee E
4 5
Enter the employee from which to start work
A
A 0 2
B 2 4
D 4 6
E 6 8
C 8 10
B 10 12
D 12 14
E 14 16
A 16 18
B 18 20
D 20 22
E 22 23
B 23 25
A 25 26
D 26 28
D 28 29
PS C:\Users\Lenovo\OneDrive\Desktop\2020ITB065_TathagataGhosh_DSA_Ass7>
```