|  |  |  |
| --- | --- | --- |
| **Name: SUNDEEP A** | **SRN:PES1UG20CS445** | **Section: O** |
| **Date:17/6/2021** | **Week Number:7** |

|  |  |
| --- | --- |
| **1** | Define a structure called cricket that will describe the following information:  player name  team name  batting average  Using cricket, declare an array player with 5 elements and write a program to read the information about all the 5 players and print a team-wise list containing names of player with their batting average.Write functions for the following:  i) Read the information of all the 5 players  ii)Sorting the players  iii)Displaying team-wise list containing names of player with their batting average  **Input:**  Enter data of 5 players  Enter PName TName BAvg for player-1 = sachin  India  98  Enter PName TName BAvg for player-2 = Rahul  India  45  Enter PName TName BAvg for player-3 = Jonty  Australia  89  Enter PName TName BAvg for player-4 = Imran  pakistan  75  Enter PName TName BAvg for player-5 = Shen  Australia  29  **Output:**  After teamwise sorting... Player list is  Jonty Australia 89.00  Shen Australia 29.00  sachin India 98.00  Rahul India 45.00  Imran pakistan 75.00 |
|  | **Program:**  **#include<stdio.h>**  **#include<conio.h>**  **#include<string.h>**  **//structure for holding players details**  **struct cricket{**  **char pname[20];**  **char tname[20];**  **float bavg;**  **};**  **typedef struct cricket crick;**  **void readp(crick \*p);**  **void teamsort(crick \*p);**  **void displayp(crick \*p);**  **int main()**  **{**  **crick player[5];**  **readp(player);**  **printf("BEFORE TEAM WISE SORTING\n\n");**  **displayp(player);**  **teamsort(player);**  **printf("AFTER TEAM WISE SORTING\n\n");**  **displayp(player);**  **return 0;**  **}**  **//Input function**  **void readp(crick \*p)**  **{**  **printf("ENTER THE DATA OF 5 PERSONS:\n");**  **for(int j=0;j<5;j++)**  **{**  **printf("ENTER PLAYERname TEAMname BATTINGaverage FOR PLAYER- %d=",j+1);**  **scanf("%s",&p[j].pname);**  **scanf("%s",&p[j].tname);**  **scanf("%f",&p[j].bavg);**  **}**  **}**  **//Sorting function based on teams name**  **void teamsort(crick \*p)**  **{**  **crick temp;**  **for(int j=0;j<5;j++)**  **{**  **for(int k=0;k<4-j;k++)**  **{**  **if(strcmp(p[k].tname,p[k+1].tname)>0)**  **{**  **temp=p[k];**  **p[k]=p[k+1];**  **p[k+1]=temp;**  **}**  **}**  **}**  **}**  **//Display function**  **void displayp(crick \*p)**  **{**  **for(int j=0;j<5;j++)**  **{**  **printf("%s\t %s\t\t %.2f\n",p[j].pname,p[j].tname,p[j].bavg);**  **}**  **}** |
|  | **Output Screenshot:** |
| **2** | **Implement Priority Queue using an Unordered Linked list.**  Write functions for the following  1)Initialization  2)Enqueue  3)Dequeue  4)Display  **Output:**  enter ua choice  1.insert 2.delete 3.display 4 exit  1  enter the detail and priority  10  1  enter ua choice  1.insert 2.delete 3.display 4 exit  1  enter the detail and priority  20  2  enter ua choice  1.insert 2.delete 3.display 4 exit  1  enter the detail and priority  30  3  enter ua choice  1.insert 2.delete 3.display 4 exit  3  30 3  20 2  10 1  enter ua choice  1.insert 2.delete 3.display 4 exit  1  enter the detail and priority  40  0  enter ua choice  1.insert 2.delete 3.display 4 exit  3  40 0  30 3  20 2  10 1  enter ua choice  1.insert 2.delete 3.display 4 exit  2  deleted node detail is 30 with priority 3  enter ua choice  1.insert 2.delete 3.display 4 exit  2  deleted node detail is 20 with priority 2  enter ua choice  1.insert 2.delete 3.display 4 exit  2  deleted node detail is 10 with priority 1  enter ua choice  1.insert 2.delete 3.display 4 exit  2  deleted node detail is 40 with priority 0  enter ua choice  1.insert 2.delete 3.display 4 exit  2  no elements to delete  enter ua choice  1.insert 2.delete 3.display 4 exit  4 |
|  | **Program:**  **#include<stdio.h>**  **#include<stdlib.h>**  **#include<string.h>**  **//structure used to store the details and priority**  **typedef struct component**  **{**  **char details[20];**  **int priority;**  **}compo;**  **//used for linking**  **struct node**  **{**  **compo c;**  **struct node \*link;**  **};**  **typedef struct node node\_t;**  **struct priority\_queue**  **{**  **node\_t \*head;**  **};**  **typedef struct priority\_queue prio\_Q;**  **void init(prio\_Q\* b);**  **void EQ(prio\_Q\* b,compo\* com);**  **void DQ(prio\_Q\* b);**  **void disp(const prio\_Q\* b);**  **int main()**  **{**  **prio\_Q q;**  **init(&q);**  **compo c;**  **printf("Enter your choice 1.insert 2.delete 3.disp 4.exit\n");**  **int choice;**  **scanf("%d",&choice);**  **do{**  **switch(choice){**  **case 1: printf("Enter data and priority : ");**  **scanf("%s %d",c.details,&c.priority);**  **EQ(&q,&c);**  **break;**  **case 2: DQ(&q);**  **break;**  **case 3: disp(&q);**  **break;**  **default: return 0;**  **}**  **printf("Enter your choice 1.insert 2.delete 3.disp 4.exit\n");**  **scanf("%d",&choice);**  **}while(choice!=4);**  **return 0;**  **}**  **//initialization**  **void init(prio\_Q\* b)**  **{**  **b->head=NULL;**  **}**  **//enqueue**  **void EQ(prio\_Q\* b,compo\* com)**  **{**  **node\_t\* temp=(node\_t\*)malloc(sizeof(node\_t));**  **strcpy(temp->c.details,com->details);**  **temp->c.priority=com->priority;**  **temp->link=b->head;**  **b->head=temp;**  **}**  **//Dequeue**  **void DQ(prio\_Q\* b)**  **{**  **if(b->head==NULL)**  **printf("NO elements to delete\n");**  **else**  **{**  **node\_t\* present=b->head;**  **node\_t\* prev=NULL;**  **int max=0;**  **node\_t\* prev\_max=NULL;**  **while(present!=NULL)**  **{**  **if(present->c.priority>=max)**  **{**  **max=present->c.priority;**  **prev\_max=prev;**  **}**  **prev=present;**  **present=present->link;**  **}**  **compo co;**  **if(prev\_max!=NULL)**  **{**  **node\_t\* temp=prev\_max->link;**  **prev\_max->link=temp->link;**  **strcpy(co.details,temp->c.details);**  **co.priority=temp->c.priority;**  **free(temp);**  **}**  **else**  **{**  **node\_t\* temp=b->head;**  **b->head=b->head->link;**  **strcpy(co.details,temp->c.details);**  **co.priority=temp->c.priority;**  **free(temp);**  **}**  **printf("Deleted node details is %s with %d priority\n",co.details,co.priority);**  **}**  **}**  **//Display**  **void disp(const prio\_Q\* b)**  **{**  **node\_t\* p=b->head;**  **if(p==NULL)**  **printf("No elements in queue\n");**  **else**  **{**  **while(p!=NULL)**  **{**  **printf("%s %d\n",p->c.details,p->c.priority);**  **p=p->link;**  **}**  **}**  **}** |
|  | **Output Screenshot:** |