|  |  |  |
| --- | --- | --- |
| **Name: Naren Chandrashekhar** | **SRN: PES2UG20CS216** | **Section: G** |
| **Date: 17/06/2021** | **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<string.h>  void read(int);  void display(int);  void sort(int);  struct cricket  {  char player\_name[30];  char team\_name[30];  float batting\_avg;  }cricinfo[5];  int main()  {  int n = 5;  read(n);  sort(n);  display(n);  return 0;  }  void read(int n)  {  int i;  printf("Enter data of 5 players \n");  for(i=0;i<n;i++)  {  printf("Enter PName TName BAvg for player-%d\t",i+1);  scanf("%s%s%f",&cricinfo[i].player\_name,&cricinfo[i].team\_name,&cricinfo[i].batting\_avg);  }  }  void sort(int n)  {  char temp[30];  float temp1;  for(int i=0;i<n-1;i++)  {  for(int j=0;j<n-i-1;j++)  {  if(strcmp(cricinfo[j].team\_name,cricinfo[j+1].team\_name)>0)  {  strcpy(temp,cricinfo[j].team\_name);  strcpy(cricinfo[j].team\_name,cricinfo[j+1].team\_name);  strcpy(cricinfo[j+1].team\_name,temp);  strcpy(temp,cricinfo[j].player\_name);  strcpy(cricinfo[j].player\_name,cricinfo[j+1].player\_name);  strcpy(cricinfo[j+1].player\_name,temp);  temp1 = cricinfo[j].batting\_avg;  cricinfo[j].batting\_avg = cricinfo[j+1].batting\_avg;  cricinfo[j+1].batting\_avg = temp1;  }  }  }  }  void display(int n)  {  int i;  printf("After teamwise sorting...the player list is\n");  for(i=0;i<n;i++)  {  printf("%s %s %.2f\n",cricinfo[i].player\_name,cricinfo[i].team\_name,cricinfo[i].batting\_avg);  }  } |
|  | **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>  void enqueue();  void dequeue();  void display();  struct node  {  int data;  int priority;  struct node\* next;  };  struct node \*front = NULL;  int main()  {  int choice;  while(1)  {  printf("Enter your choice\n1.Insert 2.Delete 3.Display 4.Exit \nYour choice: ");  scanf("%d",&choice);  switch(choice)  {  case 1: enqueue();  break;  case 2: dequeue();  break;  case 3: display();  break;  case 4: exit(0);  default: printf("Invalid input given");  }  }  return 0;  }  struct node\* newnode()  {  struct node \*temp = NULL;  temp = (struct node \*)malloc(sizeof(struct node));  printf("Enter the data and priority ");  scanf("%d %d",&temp->data,&temp->priority);  temp->next = NULL;  return temp;  }  void enqueue()  {  struct node \*temp =NULL,\*temp1=NULL;  temp = newnode();  if(front == NULL)  {  front = temp;  }  else  {  if(temp->priority <= front->priority)  {  temp->next = front;  front = temp;  }  else  {  temp1 = front;  while(temp1 != NULL)  {  if(temp->priority > temp1->priority && temp1->next != NULL)  {  temp1 = temp1->next;  }  else if(temp1->next == NULL)  {  temp1->next = temp;  temp1 = temp->next;  }  }  }  }  }  void dequeue()  {  struct node\* temp = NULL;  if(front == NULL)  printf("Empty queue\n");  else  {  temp = front;  printf("%d data with priority %d has been deleted\n",front->data,front->priority);  front = front->next;  free(temp);  }  }  void display()  {  struct node\* temp = front;  if(front == NULL)  printf("Empty queue.\n");  else  {  while(temp != NULL)  {  printf("Priority and data is %d %d\n",temp->priority,temp->data);  temp = temp->next;  }  }  } |
|  | **Output Screenshot:** |