```c
ASSIGNMENT 8
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#define COUNT 10

struct car_rent
{
  char car_name[20];
  char location[20];
  int price;
  int km;
  int time;
  struct car_rent *link;
}*start;

struct priceTree
{
  int price;
  struct priceTree *left;
  struct priceTree *right;
};
struct car_rent* getNewNode();
struct car_rent *create(struct car_rent *start);
struct car_rent *insert(struct car_rent *start,int n);
struct car_rent *del(struct car_rent *start,char item[]);
struct car_rent *sort (struct car_rent *start);
struct car_rent *modify(struct car_rent *start,char item[]);
void display (struct car_rent *start);
void *search(struct car_rent *start,char item[]);
void create_queue();
void display_queue();
void delete_queue();
struct car_rent *front=NULL;
struct car_rent *rear=NULL;
struct priceTree *node;
struct priceTree *root;
struct priceTree *getTree();
struct priceTree *getnode(int x);
struct priceTree *BalancedBst(int arr[],int s,int e);
void sortTree(int arr[],int n);
void print2DUtil(struct priceTree *root, int space);
void print2D(struct priceTree *root);
struct car_rent *p,*q,*temp;
char item[20];
int i,n,x,j,ch,l;
int s,e,point=0;
int flag=0;
void main()
{
   struct car_rent *start=NULL;

   printf("\t\t****************************************\n");
   printf("\t\t****************************************\n");
   printf("\t\t**   CAR RENTAL SYSTEM   **\n");
   printf("\t\t****************************************\n");
   printf("\t\t****************************************\n");
  while(1)
    {
     printf(" 1) to create\n 2) to display.\n 3) to insert.\n 4) to delete. \n 5) to sort by price.\n 6) to search. \n 7) to modify. \n 8) create using queue.\n 9) display using queue.\n 10) delete using queue.\n 11) Create tree of price.\n 12) exit(0).\n\n ");
     printf("Enter your choice-");
     scanf("%d",&ch);
      switch (ch)
      {
        case 1:
            start=create(start);
            break;
        case 2:
            display(start);
            break;
        case 3:
            start=insert(start,n);
            n++;
            break;
        case 4:
            printf("Enter item you want to delete:");
            scanf("%s",item);
            start=del(start,item);
            break;
        case 5:
            sort(start);
            break;
        case 6:
            printf("Enter item you want to search:");
            scanf("%s",item);
            search(start,item);
            break;
        case 7:
            printf("Enter item you want to modify:");
            scanf("%s",item);
            start=modify(start,item);
            break;
        case 8:
            create_queue();
            break;
        case 9:
            display_queue();
            break;
        case 10:
            delete_queue();
            break;
        case 11:
            print2D(getTree(start));
            break;
        case 12:
            exit(0);
            break;
        default:
            printf("error!");
    }

  }
}

struct car_rent *create(struct car_rent *start)
{

    struct car_rent *temp;
    printf("Enter no of cars you want to rent:");
    scanf("%d",&n);
    temp=start;
    for(i=0;i<n;i++)
     {
       temp=(struct car_rent*)malloc(sizeof(struct car_rent)*1);
       while(1)
       {
        printf("\tenter car name:");
        scanf("%s",(temp->car_name));
        printf("\tenter location");
        scanf("%s",(temp->location));
        l=strlen(temp->car_name);
        l=strlen(temp->location);
        if(l > 10 )
        {
          printf("\tPlease enter less characters in list name\n");
          printf("\tPlease enter less characters in list name\n");
        }
        else
        break;
       }

       do{
           printf("\tenter price:");
           scanf("%d",&(temp->price));
          if(temp->price > 1 && temp->price < 99999)
          {
            break;
          }
          else
          {
            printf("\tPlease enter less price\n");
          }
         }while(1);

       do{
          printf("\tenter time:");
          scanf("%d",&(temp->time));
            if(temp->time > 1 && temp->time < 55500)
          {
            break;
          }
          else
          {
            printf("\tPlease enter less time\n");
          }
        }while(1);

       do{

           printf("\tenter distance:");
           scanf("%d",&(temp->km));
           if(temp-> km > 0 && temp-> km < 22200)
          {
            break;
          }
          else
          {
            printf("\tSORRY..not available Please enter distance between(1 to 11111200)\n");
          }
        }while(1);
       printf("\n");

      if(start==NULL)
      {
       start=temp;
       temp->link=NULL;
      }

      else
      {
          p=start;
          while(p->link!=NULL)
          p=p->link;
          p->link=temp;
          temp->link=NULL;
      }
}
return start;
}

void display (struct car_rent *start)
{
   if(start==NULL)
   {
       printf("\n*Order List is Empty*\n\n");
       return;
   }
  p=start;
  printf("\n\tcar name\tPrice\tlocation\ttime\tdistance.\n");
  while(p!=NULL)
    {
      printf("\n\t%s\t %d\t%s\t%d\t\t%d",p->car_name,p->price,p->location,p->time,p->km);
      p=p->link;
    }
    printf("\n");
}

struct car_rent *insert(struct car_rent *start,int n)
{
  struct car_rent *temp;
  int pos,count=0;
  p=start;
  temp=(struct car_rent*)malloc(sizeof(struct car_rent)*1);
  printf("Enter location you want to insert:");
  scanf("%d",&pos);
  if(pos<n)
  {
   count++;
  }

  printf("%d\n",n );
    if(pos==1)
    {
      while(1)
       {
        printf("\tenter car name:");
        scanf("%s",(temp->car_name));
        printf("\tenter location");
        scanf("%s",(temp->location));
        l=strlen(temp->car_name);
        l=strlen(temp->location);
        if(l > 10 )
        {
          printf("\tPlease enter less characters in list name\n");
          printf("\tPlease enter less characters in list name\n");
        }
        else
           break;
       }

       do{
           printf("\tenter price:");
           scanf("%d",&(temp->price));
          if(temp->price > 1 && temp->price < 99999)
          {
            break;
          }
          else
          {
            printf("\tPlease enter less price\n");
          }
        }while(1);

       do{
          printf("\tenter time:");
          scanf("%d",&(temp->time));
            if(temp->time > 1 && temp->time < 55500)
          {
            break;
          }
          else
          {
            printf("\tPlease enter less time\n");
          }
        }while(1);
      do{
           printf("\tenter distance :");
           scanf("%d",&(temp->km));
           if(temp-> km > 0 && temp-> km < 22200)
          {
            break;
          }
          else
          {
            printf("\tSORRY..not available Please enter receipt_code between(1 to 200)\n");
          }
        }while(1);
         temp->link=start;
         start=temp;

       return start;
    }


    else{



     for(i=1;i<pos-1 && p!=NULL;i++)
        p=p->link;
        if(p==NULL)
         printf("\n*less no of items*\n\n");

    else
    {
       while(1)
       {
        printf("\tenter car name:");
        scanf("%s",(temp->car_name));
        printf("\tenter location");
        scanf("%s",(temp->location));
        l=strlen(temp->car_name);
        l=strlen(temp->location);
        if(l > 10 )
        {
          printf("\tPlease enter less characters in list name\n");
          printf("\tPlease enter less characters in list name\n");
        }
        else
           break;
       }

       do{
           printf("\tenter price:");
           scanf("%d",&(temp->price));
          if(temp->price > 1 && temp->price < 99999)
          {
            break;
          }
          else
          {
            printf("\tPlease enter less price\n");
          }
        }while(1);

       do{
          printf("\tenter time:");
          scanf("%d",&(temp->time));
          if(temp->time > 1 && temp->time < 55500)
          {
            break;
          }
          else
          {
            printf("\tPlease enter less time\n");
          }
        }while(1);
      do{
           printf("\tenter distance :");
           scanf("%d",&(temp->km));
           if(temp-> km > 0 && temp-> km < 22200)
          {
            break;
          }
          else
          {
            printf("\tSORRY..not available Please enter receipt_code between(1 to 200)\n");
          }
        }while(1);
        temp->link=p->link;
         p->link=temp;
}
   return start;
    }
}

struct car_rent *del(struct car_rent *start,char item[])
{
  struct car_rent *temp;
  if(strcmp(start->car_name,item)==0)
  {
      temp=start;
      start=temp->link;
      free(temp);
      return start;
  }
      p=start;

      while(p->link!=NULL)
      {
        if(strcmp(p->link->car_name,item)==0)
          {
            temp=p->link;
            p->link=temp->link;
            free(temp);
            return start;
          }
         p=p->link;
     }
}

struct car_rent *sort (struct car_rent *start)
{
   struct car_rent *temp;

    temp=(struct car_rent*)malloc(sizeof(struct car_rent)*1);
    for(p=start;p->link!=NULL;p=p->link)
    {
      for(q=p->link;q!=NULL;q=q->link)
      {
        if(p->price > q->price)
          {
             temp->price=p->price;
             p->price=q->price;
             q->price=temp->price;
             temp->time=p->time;
             p->time=q->time;
             q->time=temp->time;
             temp->km=p->km;
             p->km=q->km;
             q->km=temp->km;
             strcpy(temp->car_name,p->car_name);
             strcpy(p->car_name,q->car_name);
             strcpy(q->car_name,temp->car_name);
             strcpy(temp->location,p->location);
             strcpy(p->location,q->location);
             strcpy(q->location,temp->location);
          }
      }
    }
}

void *search(struct car_rent *start,char item[])
{
      if(strcmp(start->car_name,item)==0)
         {
           printf("\nITEM FOUND\n");
           printf("\ncar name\tPrice\tlocation\ttime\tdistance\n.");
           printf("\n%s\t %d\t %s\t %d\t %d\n",start->car_name,start->price,start->location,start->time,start->km);
           flag++;
         }
       p=start;
       while(p->link!=NULL)
        {
            if(strcmp(p->link->car_name,item)==0)
          {
           printf("\n*ITEM FOUND\n");
           printf("\ncar name\tPrice\tlocation\ttime\tdistance");
           printf("\n%s\t %d\t %s\t %d\t %d\n",p->link->car_name,p->link->price,p->link->location,p->link->time,p->link->km);
          }
           p=p->link;
           flag++;
        }

        if(flag==0)
        printf("\n\nnot found\n\n");

}

struct car_rent *modify(struct car_rent *start,char item[])
{
    char ans1,ans2;
    int chw=0;
    if(strcmp(start->car_name,item)==0)
     {
       printf("\nITEM FOUND\n");
       printf("\ncar name\tPrice\tlocation\ttime\tdistance\n.");
       printf("\n%s\t %d\t %s\t %d\t %d\n",start->car_name,start->price,start->location,start->time,start->km);

     do
     {
       printf("enter:\n 1) to modify car name\n 2) to modify list price:");
       scanf("%d",&ch);
       switch (ch)
       {
       case 1:
             while(1)
             {
              printf("\tenter product name:");
              scanf("%s",(temp->car_name));
              l=strlen(temp->car_name);
              if(l > 10 )
              {
                printf("\tPlease enter less characters in list name\n");
              }
              else
              {
                 break;
              }
             }
              break;

       case 2:
              do{
                 printf("\tenter price:");
                 scanf("%d",&(temp->price));
                if(temp->price > 1 && temp->price < 99999)
                {
                  break;
                }
                else
                {
                  printf("\tPlease enter less price\n");
                }
              }while(1);
              break;

       }

       printf("press 1 to continue modifying:");
       scanf("%d",&chw);
      }while(chw==1);

            flag++;
            return start;
      }

       p=start;

         while(p->link!=NULL)
         {
            if(strcmp(p->link->car_name,item)==0)
          {
           printf("\n*ITEM FOUND\n");
             printf("\ncar name\tPrice\tlocation\ttime\tdistance");
           printf("\n%s\t %d\t %s\t %d\t %d\n",p->link->car_name,p->link->price,p->link->location,p->link->time,p->link->km);

            do
            {
              printf("enter:\n 1) to modify car name\n 2) to modify list price:");
              scanf("%d",&ch);

          switch (ch)
          {
             case 1:
                     while(1)
                     {
                      printf("\tenter car name:");
                      scanf("%s",(temp->car_name));
                      l=strlen(temp->car_name);
                      if(l > 10 )
                      {
                        printf("\tPlease enter less characters in list name\n");
                      }
                      else
                      {
                         break;
                      }
                     }
                    break;

             case 2:
                   do{
                       printf("\tenter price:");
                       scanf("%d",&(temp->price));
                      if(temp->price > 1 && temp->price < 99999)
                      {
                        break;
                      }
                      else
                      {
                        printf("\tPlease enter less price\n");
                      }
                    }while(1);
                   break;

          }
            printf("press 1 to continue modifying:");
            scanf("%d",&chw);
            }while(chw==1);


           }
               p=p->link;
               flag++;
               return start;
        }

        if(flag==0)
        printf("\n\nnot found\n\n");
}

struct car_rent* getNewNode()
{
    return (struct car_rent*)malloc(sizeof(struct car_rent));
}
void create_queue()
{
  int q;
  struct car_rent *temp;
  printf("Enter no.of type of item:");
  scanf("%d",&q);
```

```
   for(i=0;i<n;i++)
     {
       temp=(struct car_rent*)malloc(sizeof(struct car_rent)*1);
       while(1)
       {
        printf("\tenter car name:");
        scanf("%s",(temp->car_name));
        printf("\tenter location");
        scanf("%s",(temp->location));
        l=strlen(temp->car_name);
        l=strlen(temp->location);
        if(l > 10 )
        {
          printf("\tPlease enter less characters in list name\n");
          printf("\tPlease enter less characters in list name\n");
        }
        else
        break;
       }

       do{
           printf("\tenter price:");
           scanf("%d",&(temp->price));
          if(temp->price > 1 && temp->price < 99999)
          {
            break;
          }
          else
          {
            printf("\tPlease enter less price\n");
          }
         }while(1);

       do{
          printf("\tenter time:");
          scanf("%d",&(temp->time));
            if(temp->time > 1 && temp->time < 55500)
          {
            break;
          }
          else
          {
            printf("\tPlease enter less time\n");
          }
        }while(1);

       do{

           printf("\tenter distance:");
           scanf("%d",&(temp->km));
           if(temp-> km > 0 && temp-> km < 22200)
          {
            break;
          }
          else
          {
            printf("\tSORRY..not available Please enter distance between(1 to 11111200)\n");
          }
        }while(1);
        printf("\n");

        if (rear==NULL)
        {
            rear= temp;
            front = temp;
            rear->link=NULL;
            continue;
        }
        rear->link=temp;
        rear=temp;
     }
}
void display_queue()
{
    if(front==NULL)
    {
        printf("\n No ITEM IN LIST\n");
    }
    else
    {
        p=front;
        printf("\n\tcar name \tPRICE \tlocation \ttime \tdistance \n\n");
        while(p!=NULL)
           {
            printf("\t%s\t%d\t%s\t%d\t%d",p->car_name,p->price,p->location,p->time,p->km);
            printf("\n");
            p=p->link;
           }
    }
}
void delete_queue()
{
    if(front==NULL)
    {
        printf("\n*Queue Underflow*\n");
    }
    else
    {
      temp=front;
      front=front->link;
      free(temp);
    }
}
struct priceTree *getTree(struct car_rent *start)
{
  int arr[n];
  struct car_rent *temp=start;

  while(temp!=NULL)
  {
    arr[point++]=temp->price;
    temp=temp->link;

  }
    sortTree(arr,n);
    return BalancedBst(arr,0,n-1);
}
void sortTree(int arr[],int n)
{
  int temp;
  for(int i = 0; i < n; i++)
    {
        for(int j = 0; j < (n-1-i); j++)
        {
            if(arr[j] > arr[j+1]) //for decending order change "arr[j]<arr[j+1]"
            {
                temp = arr[j];
                arr[j] = arr[j+1];
                arr[j+1] = temp;
            }
        }
    }
}
struct priceTree *getnode(int x)
{
  struct priceTree *temp=(struct priceTree*)malloc(sizeof(struct priceTree));
  temp->price=x;
  temp->left=temp->right=NULL;
  return temp;
}
struct priceTree *BalancedBst(int arr[],int s,int e)
{
  if(s > e)
    return NULL;

  int mid=(s+e)/2;
  struct priceTree *root=getnode(arr[mid]);
  root->left=BalancedBst(arr,s,mid-1);
  root->right=BalancedBst(arr,mid+1,e);
  return root;
}

void print2DUtil(struct priceTree *root, int space)
{

    if (root == NULL)
        return;
    space+=COUNT;
    print2DUtil(root->right, space);

    printf("\n");
    for (int i = COUNT; i < space; i++)
        printf(" ");
    printf("%d\n\n", root->price);

    print2DUtil(root->left, space);
}

```

```
void print2D(struct priceTree *root)
{
    print2DUtil(root, 0);
}














/*

OUTPUT
                ****************************************
                ****************************************
                **   CAR RENTAL SYSTEM   **
                ****************************************
                ****************************************
 1) to create
 2) to display.
 3) to insert.
 4) to delete.
 5) to sort by price.
 6) to search.
 7) to modify.
 8) create using queue.
 9) display using queue.
 10) delete using queue.
 11) Create tree of price.
 12) exit(0).

 Enter your choice-2

*Order List is Empty*

 1) to create
 2) to display.
 3) to insert.
 4) to delete.
 5) to sort by price.
 6) to search.
 7) to modify.
 8) create using queue.
 9) display using queue.
 10) delete using queue.
 11) Create tree of price.
 12) exit(0).

 Enter your choice-1
Enter no of cars you want to rent:2
        enter car name:audi
        enter locationpune
        enter price:23
        enter time:34
        enter distance:64

        enter car name:bmw
        enter locationmumbai
        enter price:4225
        enter time:43
        enter distance:53

 1) to create
 2) to display.
 3) to insert.
 4) to delete.
 5) to sort by price.
 6) to search.
 7) to modify.
 8) create using queue.
 9) display using queue.
 10) delete using queue.
 11) Create tree of price.
 12) exit(0).

 Enter your choice-2

```

```
        car name        Price   location        time    distance.

        audi     23     pune    34              64
        bmw      4225   mumbai  43              53
 1) to create
 2) to display.
 3) to insert.
 4) to delete.
 5) to sort by price.
 6) to search.
 7) to modify.
 8) create using queue.
 9) display using queue.
 10) delete using queue.
 11) Create tree of price.
 12) exit(0).

 Enter your choice-3
Enter location you want to insert:1
2
        enter car name:maruti
        enter locationsonari
        enter price:2424
        enter time:24
        enter distance :522
 1) to create
 2) to display.
 3) to insert.
 4) to delete.
 5) to sort by price.
 6) to search.
 7) to modify.
 8) create using queue.
 9) display using queue.
 10) delete using queue.
 11) Create tree of price.
 12) exit(0).

 Enter your choice-2

        car name        Price   location        time    distance.

        maruti   2424   sonari  24              522
        audi     23     pune    34              64
        bmw      4225   mumbai  43              53
 1) to create
 2) to display.
 3) to insert.
 4) to delete.
 5) to sort by price.
 6) to search.
 7) to modify.
 8) create using queue.
 9) display using queue.
 10) delete using queue.
 11) Create tree of price.
 12) exit(0).

 Enter your choice-4
Enter item you want to delete:bmw
 1) to create
 2) to display.
 3) to insert.
 4) to delete.
 5) to sort by price.
 6) to search.
 7) to modify.
 8) create using queue.
 9) display using queue.
 10) delete using queue.
 11) Create tree of price.
 12) exit(0).

 Enter your choice-2

        car name        Price   location        time    distance.

        maruti   2424   sonari  24              522
        audi     23     pune    34              64

 Enter your choice-5
 1) to create
 2) to display.
 3) to insert.
 4) to delete.
 5) to sort by price.
 6) to search.
 7) to modify.
 8) create using queue.
 9) display using queue.
 10) delete using queue.
 11) Create tree of price.
 12) exit(0).

 Enter your choice-2

        car name        Price   location        time    distance.

        audi     23     pune    34              64
        maruti   2424   sonari  24              522
 1) to create
 2) to display.
 3) to insert.
 4) to delete.
 5) to sort by price.
 6) to search.
 7) to modify.
 8) create using queue.
 9) display using queue.
 10) delete using queue.
 11) Create tree of price.
 12) exit(0).

 Enter your choice-6
Enter item you want to search:audi

ITEM FOUND

car name        Price   location        time    distance
.
audi     23      pune    34      64
 1) to create
 2) to display.
 3) to insert.
 4) to delete.
 5) to sort by price.
 6) to search.
 7) to modify.
 8) create using queue.
 9) display using queue.
 10) delete using queue.
 11) Create tree of price.
 12) exit(0).

1) to create
 2) to display.
 3) to insert.
 4) to delete.
 5) to sort by price.
 6) to search.
 7) to modify.
 8) create using queue.
 9) display using queue.
 10) delete using queue.
 11) Create tree of price.
 12) exit(0).

 Enter your choice-11

          2424


23
*/

```