**SORTING MECHANISM**

|  |  |
| --- | --- |
| 1 | Write a program for Merge sort |
|  | main()  {  int x[8]={3,9,2,10,40,1,5,13};  int i;  split(x,0,7);  for(i=0;i<8;i++)  {  printf("%d ",x[i]);  }  }  int split(int x[],int low,int up)  {  int m;  if(low<up)  {  m=(low+up)/2;  split(x,low,m);  split(x,m+1,up);  merge(x,low,m,up);  }  }  int merge(int x[],int low,int m,int up)  {  int i=low,j=m+1,k=low;  int z[8];  while(i<=m && j<=up)  {  if(x[i]<x[j])  {  z[k]=x[i];  k++;  i++;  }  else  {  z[k]=x[j];  k++;  j++;  }  }  while(i<=m)  {  z[k]=x[i];  k++;  i++;  }  while(j<=up)  {  z[k]=x[j];  k++;  j++;  }  for(i=low;i<=up;i++)  {  x[i]=z[i];  }  } |
| 2 | Write a program for Heap sort |
|  | #include<stdio.h>    void create(int []);  void down\_adjust(int [],int);    void main()  {  int heap[30],n,i,last,temp;  printf("Enter no. of elements:");  scanf("%d",&n);  printf("\nEnter elements:");  for(i=1;i<=n;i++)  scanf("%d",&heap[i]);    //create a heap  heap[0]=n;  create(heap);    //sorting  while(heap[0] > 1)  {  //swap heap[1] and heap[last]  last=heap[0];  temp=heap[1];  heap[1]=heap[last];  heap[last]=temp;  heap[0]--;  down\_adjust(heap,1);  }    //print sorted data  printf("\nArray after sorting:\n");  for(i=1;i<=n;i++)  printf("%d ",heap[i]);  }    void create(int heap[])  {  int i,n;  n=heap[0]; //no. of elements  for(i=n/2;i>=1;i--)  down\_adjust(heap,i);  }    void down\_adjust(int heap[],int i)  {  int j,temp,n,flag=1;  n=heap[0];    while(2\*i<=n && flag==1)  {  j=2\*i; //j points to left child  if(j+1<=n && heap[j+1] > heap[j])  j=j+1;  if(heap[i] > heap[j])  flag=0;  else  {  temp=heap[i];  heap[i]=heap[j];  heap[j]=temp;  i=j;  }  }  } |
| 3 | Write a program for Quick sort |
|  | main()  {  int x[8]={10,2,90,70,5,7,100,15};  int i;  quick\_sort(x,0,7);  for(i=0;i<8;i++)  printf("%d ",x[i]);  }  int quick\_sort(int x[],int low,int up)  {  int m;  if(low<up)  {  m=split(x,low,up);  quick\_sort(x,low,m);  quick\_sort(x,m+1,up);  }  }  int split(int x[],int left,int right)  {  int i,j,temp,p;  i=left;  j=right;  if(i<j)  {  p=(i+j)/2;  while(x[p]>x[i])  i++;  if(x[p]<x[i])  {  temp=x[i];  x[i]=x[p];  x[p]=temp;  p=i;  }  while(x[p]<x[j])  j--;  if(x[p]>x[j])  {  temp=x[j];  x[j]=x[p];  x[p]=temp;  p=j;  }  quick\_sort(x,left,p);  quick\_sort(x,p+1,right);  }  return p;  } |
|  |  |