MINHEAP USING C

1.create_minheap
2.getMini
3.heapsort
4.extract min
5.extractkey
6.print
7.insert

8.exit

```
#include<stdio.h>
#include<stdlib.h>
int leftchild(int i)
    {
        return(2*i+1);
    }
int rightchild(int i)
    {
        return(2*i+2);
    }
int parent(int i)
    {
        if(i%2==1)
        {
            return(i/2);
        }
        else
        {
           return((i/2)-1);
        }
    }
```

```
void swap(int* a,int* b)
   {
      int t=*a;
      *a=*b;
      *b=t;
   }
void min_heapify(int arr[],int i,int n)
   {   printf("\n********--------\n");
      int l=leftchild(i);
      int r=rightchild(i);
      int least=i;
      if(arr[i]>arr[l] && l<n)
         {
            least=l;
         }
      if(arr[least]>arr[r] && r<n)
         {
            least=r;
         }

      if(least!=i)
         {
            swap(&arr[least],&arr[i]);
            min_heapify(arr,least,n);
         }
   }
```

```
void create_minheap(int arr[],int n)
   {
      int i;
      int heapsize;

      for(i=(n/2)-1;i>=0;i--)
       {
          min_heapify(arr,i,n);
       }
   }

void heapsort(int arr[],int n)
   {
       int i;
       create_minheap(arr,n-1);

       for(i=n-1;i>=0;i--)
          {
              swap(&arr[0],&arr[i]);
              min_heapify(arr,0,i);

          }

   }
```

```
int getMini(int arr[])
{
   return(arr[0]);
}
void extractMin(int arr[],int n)
   {
      swap(&arr[0],&arr[n-1]);
      n=n-1;
      create_minheap(arr,n-1);
      print(arr,n);



   }
void decreasekey(int arr[],int n,int key,int index)
   {  int i;
      if(index<n)
      {  if(arr[index]>key)
      {



         arr[index]=key;
         i=parent(index);
         while(arr[i]> arr[index])
         {
            min_heapify(arr,i,n);
            index=i;
```

```
            i=parent(i);
        }
    }
    }
    else
    {
        printf("\n index is outof bound:\n");
    }
  }
void insert(int arr[],int* n,int data)
  {
    int i, index;
    index=*n;
    arr[index]=data;
    *n=*n+1;

     if(index < *n)
    {      i=parent(index);
        while(arr[i]> arr[index])
        {   printf("\n********** %d  %d  \n",arr[index],arr[i]);
            min_heapify(arr,i,*n);
            index=i;
            i=parent(index);
        }

    }
```

```
    }
void delet(int arr[],int p,int n)
    {



    }
void print(int arr[],int n)
    {
        int i;
        for(i=0;i<n;i++)
        {
            printf("%d\t",arr[i]);
        }
        printf("\n");
    }
int main()
    {

        int arr[50];
        int i,time=0;
        int getmin,n;
        int choice,key,index,data,ind;
        printf("\n Enter the total no of element of the
array:\n");
```

```
        scanf("%d",&n);
        printf("enter the element of the array:\n");
        for(i=0;i<n;i++)
        {
            scanf("%d",&arr[i]);
        }

        while(time!=30)
        {

            printf("Press: 0 create_minheap: 1 for getmin: 2
heapsort : 3 extractMin : 4 print: 5 decreasekey :6 insert :7
delet :8 exit :\n ");
            printf("\n Enter the choice:\n");
            scanf("%d",&choice);
            switch(choice)
            {
                case 0:
                    create_minheap(arr,n);
                    break;
                case 1:
                    getmin=getMini(arr);
                    printf("\nThe value of the getmini
is:%d\n",getmin);
                    break;
                case 2:
```

```
                heapsort(arr,n);
                break;
            case 3:
                extractMin(arr,n);
                n=n-1;
                break;
            case 4:
                print(arr,n);
                break;
            case 5:
                printf("enter the decrease key: and index:\n");
                scanf("%d %d",&key,&index);
                decreasekey(arr,n,key,index);
                break;
            case 6:
               printf("\nInsert the data:\n");
               scanf("%d",&data);
               insert(arr,&n,data);
               break;
            case 7:
               printf("enter the index which you want to
delet:\n");
               scanf("%d",&ind);
               delet(arr,ind,n);
               n=n-1;
               break;
```

```
            case 8:
                exit(0);
                break;
            default:
                break;
        }

        time++;
    }

    return 0;
}
```