/\*\*

NAME: OKAFOR ECHEZONA ELVIS

REG NO: 2012/181569

COURSE: COS461

DEPT: COMPUTER SCIENCE

\*\*/

package heap;

import java.util.Arrays;

import java.util.Random;

public class Heap {

public static int heapSize;

public static void main(String[] args) {

Random r=new Random();

int A[] = new int[2];

int B[] = new int[10];

int C[] = new int[A.length+B.length];

System.out.print("The first 10 random numbers are ");

for(int i=9; i>=0; i--)

{

B[i]=r.nextInt(10)+1; //generation of ten random numbers

System.out.print(B[i]+ " ");

}

HEAP\_SORT(B);

System.out.print("\n\n");

System.out.print("The first 10 heapified numbers are ");

System.out.println(Arrays.toString(B));

System.out.print("\n\n");

System.out.print("The additional two random numbers are ");

for(int i=0; i<2; i++)

{

A[i]=r.nextInt(10)+1;//generation of two random numbers

System.out.print(A[i]+ " ");

}

System.out.print("\n\n");

System.arraycopy(A, 0, C, 0, A.length);

System.arraycopy(B, 0, C, A.length, B.length);

HEAP\_SORT(C);

System.out.print("The 12 heapified numbers are ");// heapify the addition of 10 and 2 numbers

System.out.println(Arrays.toString(C));

System.out.print("\n");

System.out.print("The extracted items in max order are ");

for(int i =11; i>=0; i--)

{

Arrays.sort(C);

int max=C[C.length-1];

//System.out.print(max);

System.out.print(C[i]+ " ");//Using heap\_extract\_max to extract items in order

//System.out.print("\n");

}

}

public static int LEFT(int i)

{

return 2\*i+1;

}

public static int RIGHT(int i)

{

return 2\*i+2;

}

public static void BUILD\_HEAP(int A[])

{

heapSize = A.length;//heap size initialised

for(int i=A.length/2; i>=0;i--)//since n/2, n/2+1 ... are leaves we can start from n/2 step downwards

{

//call HEAPIFY on each root node starting from n/2

HEAPIFY(A, i);

}

}

public static void HEAPIFY(int A[],int i)

{

int l=LEFT(i);

int r=RIGHT(i);

int largest = -1;//index can't be negative so initialise largest index

if(l<heapSize && A[l]>A[i]){

largest = l;

}

else //if max heap property is not violated copy the root's index in largest

{

largest=i;

}

if(r<heapSize && A[r]>A[largest])

{

largest = r;

}

if(largest!=i)

{

int temp = A[i];

A[i]=A[largest];

A[largest]=temp;

HEAPIFY(A, largest);

}

}

public static void HEAP\_SORT(int A[])

{

BUILD\_HEAP(A);

for(int i=A.length-1;i>=0;i--)

{

int temp = A[0];

A[0]=A[i];

A[i]=temp;

heapSize = heapSize-1;

HEAPIFY(A,0);

}

}

}