**Name Surname =** Hasan Hüseyin Pay

**Student Number =** 2014510058

**HEAPSORT AND QUICKSORT**

In this lab session, you will practice Heapsort and Quicksort.

**Task – 1**

Below you wil find the java code of the classes Max\_Heap, Quicksort and Test. Copy this code to Eclips IDE and fill the missing parts with correct statements. (missing parts are green)

**Quicksort Class**

**public** **class** Quicksort {

**void** Quicksort(**int**[] A, **int** p, **int** r)

{

**if**(p<r)

{

**int** q = Partition(A,p,r);

//fill here

//fill here

Quicksort(A, p, q - 1);

Quicksort(A, q + 1, r);

}

}

**int** Partition(**int**[] A, **int** p, **int** r)

{

**int** pivot = A[r];

**int** i = p - 1;

**for** (**int** j = p; j<= r-1; j++ )

{

**if**(A[j] <= pivot)

{

i = i + 1;

**int** temp = A[i];

A[i] = A[j];

A[j] = temp;

}

}

//fill here

//fill here

//fill here

**int** temp = A[i+1];

A[i + 1] = A[r];

A[r] = temp;

**return** i+1;

}

**public** **void** List\_All\_Elemets(**int**[] A)

{

**for**(**int** i=0;i<A.length;i++)

{

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

}

System.***out***.println();

}

}

**Max\_Heap Class**

**public** **class** Max\_Heap

{

**int** Array[];

**int** heap\_size;

**public** **int** Parent(**int** i)

{

//fill here

**return** i % 2 == 0 ? i / 2 -1 : i/2;

}

**public** **int** Left(**int** i)

{

**return** 2\*i + 1;

}

**public** **int** Right(**int** i)

{

**return** 2\*i + 2;

}

**public** Max\_Heap(**int**[] Array)

{

heap\_size=0;

**this**.Array=Array;

}

**public** **void** Max\_Heapify(**int** i)//Maintaining the heap property

{

**int** largest;

**int** left=Left(i);

**int** right=Right(i);

// finding the index of largest among parent and children

**if**(left<=heap\_size-1 && Array[left]>Array[i])

largest=left;

**else**

largest=i;

**if**(right<=heap\_size-1 && Array[right]>Array[largest])

largest=right;

**if**(largest!=i)

{

// fill here

// fill here

// fill here

// fill here

**int** temp = Array[i];

Array[i] = Array[largest];

Array[largest] = temp;

Max\_Heapify(largest);

}

}

**public** **void** Build\_Max\_Heap()

{

heap\_size=Array.length;

**for**(**int** i=Array.length/2; i>=0; i--)

// fill here

Max\_Heapify(i);

}

**public** **void** HeapSort()

{

**int** temp;

Build\_Max\_Heap();

**for**(**int** i=Array.length-1;i>=1;i--)

{

temp = Array[0];

Array[0] = Array[i];

Array[i] = temp;

heap\_size = heap\_size - 1;

//fill here

Max\_Heapify(0);

}

}

**public** **void** List\_All\_Elemets()

{

**for**(**int** i=0;i<Array.length;i++)

{

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

}

System.***out***.println();

}

}

**Test Class**

**public** **class** Test

{

**public** **static** **void** main(String[] args)

{

**int**[] Array1={11, 13, 2, 8, 40, 100, 5, 15, 35, 11, 2, 35, 40};

**int**[] Array2={11, 13, 2, 8, 40, 100, 5, 15, 35, 11, 2, 35, 40};

Max\_Heap max\_heap1=**new** Max\_Heap(Array1);

max\_heap1.HeapSort();

Quicksort quicksort1 = **new** Quicksort();

quicksort1.Quicksort(Array2, 0, Array2.length - 1);

max\_heap1.List\_All\_Elemets();

quicksort1.List\_All\_Elemets(Array2);

}

}

**Task – 2**

Copy your codes below. Make your own statements red and upload this document.