Doni Andrian 6182001020

EXC 1

```
Input: -
Output: element with the smallest priority
getMin()
    if (isEmpty) return null
    else
        return A[1]
input: -
output : the element with the smallest priority
removeMin()
    if (isEmpty) return null
    else
        temp = A[1]
        for I = 2 to size do
            A[i-1] = A[i]
        size = size – 1
        return temp
input:elemen
output : -
enqueue(e)
    if size < A.length
        size = size + 1
        A[size] = e
    A.sort
Input : -
Output : true/false
Empty()
    Return size == 0
```

EXC 2

Operation	**Running Time: PQ with unordered array**	**Running Time: PQ with ordered array**
getMin	O(n)	O(1)
removeMin	O(n)	O(n)
enqueue	O(1)	O(n log n)
isEmpty	O(1)	O(1)

EXC 3

Input: element

Output : hasil reversenya

```
Reverse(e)
        While (!stack.empty())
                Queue.push(s.top())
                Stack.pop()
        While (!queue.empty)
```

EXC 4

Stack:

Input: kata

Output: palindrome atau tidak

```
Checkpalindrome()
        For I = 1 to size of kata
                stack.push(kata.char at I)
        Deklarasi reverse sebagai string kosong
        While (!stack.isEmty())
                reverse += stack.pop()
        if( reverse == kata) return “palindrome”
        else
                return “tidak palindrome”
```

Queue:

Input: kata

Output: palindrome atau tidak

```
Checkpalindrome()
    for I = kata.length -1 downto 0
        queue.add(kata.charAt I)
    Deklarasi Reverse sebagai string kosong
    While (!queue.isEmpty())
        Reverse += queue.remove();
    if( reverse == kata) return “palindrome”
    else
        return “tidak palindrome”
```

EXC 5

```
Push()
If (qeueu1.isEmpty())
    Queue1.enqueue(E)
Else
    For I = 1 to size of queue1
        Queue2.enqueue(q1.dequeue())
    For j = 1 to size of queue1
        Queue1.enqueue(queue2.dequeue())
Pop()
    Qeueue1.dequeue()
```

EXC 6

```
Enqeue()
    Push elemen stack 1 ke stack 2
    Stack1.push(e)
```

```
Dequeue()
        If(stack1.isempty) return null
        Else
                While (!stack1.isempty)
                         Pop  = stack1.pop()
                         Stack2.push(pop)
        Return stack2.pop()
```