Priority Queue

By definition, priority queue is a data structure where each item has some kind of priority value and the item with highest priority (or lowest priority) is at the front of the queue. That means that items are removed based on their priority value. Here are some ways where you can setup the comparing method(s) for each of the language you may be using.

Java

Assuming our elements will be objects of type Node and it implements Comparable:

```
public class Node implements Comparable<Node> {

	int value;

	// constructors and methods here as needed

	public Node(int v) {
		value = v;
	}

	@Override
	public int compareTo(Node n) {

		if(value < n.value)
			return 1;
		else if(bound > n.bound)
			return -1;

		return 0;
	}
}

// To create empty priority queue using builtin PriorityQueue
PriorityQueue<Node> pQueue = new PriorityQueue<Node>();

// insert element
pQueue.add(new Node(111));

// remove element
Node n = pQueue.remove();
```

For more information: https://docs.oracle.com/javase/7/docs/api/java/util/PriorityQueue.html

C++

Assuming our elements will be objects of type Node:

```
#include <iostream>
#include <queue>
using namespace std;

class Node
{
private:
        int value;

public:
        Node(int v) : value (v) { };
        int Val() const { return value; }
};

struct CompareNode : public std::binary_function<Node*, Node*, bool>
{
  bool operator()(const Node* lhs, const Node* rhs) const
  {
    return lhs->Val() < rhs->Val();
  }
};

// create empty queue using existing priority_queue
priority_queue<Node*,vector<Node*>, CompareNode > pq;

// insert element
pq.push( new Node( 111 ) );

// remove element
Node* n = pq.top();
pq.pop();
```

For more information: http://www.cplusplus.com/reference/queue/priority_queue/

Python

Assuming our elements will be objects of type Node:

```
# Python 3 code uses existing queue.PriorityQueue
import queue

class Node(object):
```

```
    def __init__(self, value):
        self.value = value
        return

       def __lt__(self, other):
        return (other.value < self.value)

    def __eq__(self, other):
        return (self.value == other.value)

# create empty queue
q = queue.PriorityQueue()

# insert element
q.put( Node(111) )

# remove element
node = q.get()
```

For more information: https://docs.python.org/3/library/queue.html

C#

For C# there is no builtin priority queue that can be used and no standard way to do it. There are a number of implementations that can be found online and tweaked to work as one desires. Here is an article with explained code that can be downloaded for a generic priority queue: https://www.codeproject.com/Articles/126751/Priority-queue-in-C-with-the-help-of-heap-data-str