Assignment 1 Solution

(Please note that the pseudo code conventions in the question were incorrect at certain places which I have corrected):

```
Algorithm push(s, o):
if top = N then
    indicate that a stack-full error has occurred
top ← top+1
s[top] ← o

Algorithm pop(s):
if top = 0 then
    indicate that a stack-empty error has occurred
e ← s[top]
s[top] ← NULL
top ← top - 1
return e

Algorithm enqueue(q, o):
if rear = N then
    indicate that a queue-full error has occurred
    return
rear ← rear+1
q[rear] ← o
if front = 0 then
    front = 1

Algorithm dequeue(q):
if front = 0 then
    indicate that a queue-empty error has occurred
    return NULL
e ← q[front]
q[front] ← NULL
if front = rear then
    front ← 0
    rear ← 0
else
    front ← front +1
return e
```

```
Algorithm findhighestscorer(N, Names, Marks):

for i ← 1 to N do
    push(s, Marks [i]) //s is a stack
    enqueue(q1, Names [i]) //q1 is a queue
```

```
// The logic to solve this problem is to ensure that scores (stack s)
// and scorers(queue q1) can be retrieved in the same order. To achieve
// this we need to reverse the elements of the stack s and the retrieve
// elements from s & q1. The steps are summarized below:

// Step 1) Validation: If no input is specified, throw an error and return NULL

// Step 2) Optimization: Check if there is only one name & one score. In this case
skip all the steps and return the only name as the highest score

// Step 3) We create a new queue q2 & populate it by popping out the scores
// from the stack s.

// Step 4) We dequeue all the elements in q2 & push it to stack s in
// the same order. This reverses the order of elements in the stack
// s. The stack s now has scores in the same order as the names in the
// queue q1.

// Step 5) We iterate N times and dequeue names from q1 and pop scores
// from s in the same "transaction" comparing scores to find the
// highest score & the highest scorer. Note: Minor optimization to iterate from the
second element

// Step 1
if N = 0 then
      indicate an error saying nothing to calculate as it is an empty input
      return NULL

// Step 2
if N = 1 then
      return dequeue(q1) //Return the only element as the highest scorer

// Step 3
for i ← 1 to N do
    enqueue(q2,pop(s)) //q2 is a queue of marks which is taken from stack s

// Step 4
for i ← 1 to N do
    push(s, dequeue(q2)) // push to stack s from q2 to "reverse" stack s

// Step 5
highestScore ← pop(s) // Assign the first score as the highest score
highestScorer ← dequeue(q1) // Assign the first name as the highest scorer

// Iterate N -1 times to find the highest score & scorer
for i ← 2 to N do
    score ← pop(s)
    scorer ← dequeue(q1)
    if score > highestScore then
        highestScore ← score
        highestScorer ← scorer
return highestScorer

// End of Algorithm Pseudo code
```

Testcase 1:

	Stack s	Queue q1	Queue q2
Initial State	80 40 50	Rahul Sehwag Sachin	Empty
After Step 3	Empty	Rahul Sehwag Sachin	80 40 50
After Step 4	50 40 80	Rahul Sehwag Sachin	Empty

After Step 5 - Returns highest scorer as Sachin

Testcase 2:

	Stack s	Queue q1	Queue q2
Initial State	12 13 14 11	Ricky Surya Steve Akram	Empty
After Step 3	Empty	Ricky Surya Steve Akram	12 13 14 11
After Step 4	11 14 13 12	Ricky Surya Steve Akram	Empty

After Step 5 - Returns highest scorer as Surya

Testcase 3:

	Stack s	Queue q1	Queue q2
Initial State	12	Saurav	Empty
After Step 2	12	Empty	Empty

After Step 2 - Returns highest scorer as Saurav

Testcase 4:

	Stack s	Queue q1	Queue q2
Initial State	Empty	Empty	Empty

After Step 1 - Returns NULL & throws an error saying input is invalid