Cody Problem 16. Return the largest number that is adjacent to a zero

This example comes from Steve Eddins' blog: Learning Lessons from a One-Liner

Write a function that takes a list or array of numbers as input and return the largest number that is adjacent to a zero.

Example:

```
% Input  x = [1 5 3 0 2 7 0 8 9 1 0]
% Output y is 8
```

This problem was originally posed by Greg Wilson of Software Carpentry.

Scratch Pad

```
x = [1 5 3 0 2 7 0 8 9 1 0]
```

```
x = 1×11
     1     5     3     0     2     7     0     8     9     1     0
```

```
nearZero(x)
```

```
ans = 8
```

```
x = [0 9 3 0 2 7 0 8 9 1 0]
```

```
x = 1×11
     0     9     3     0     2     7     0     8     9     1     0
```

```
nearZero(x)
```

```
ans = 9
```

```
x = [0]
```

```
x = 0
```

```
nearZero(x)
```

```
ans =

     []
```

```
x = [7 0 8 0 9 0]
```

```
x = 1×6
     7     0     8     0     9     0
```

```
nearZero(x)
```

```
ans = 9
```

Solution

```
function y = nearZero(x)
    % Find indices of all zeros in the array
    zeroIndices = find(x == 0);

    % Initialize a list to store adjacent numbers
    adjacentNumbers = [];

    % Iterate through each zero index
    for i = 1:length(zeroIndices)
        index = zeroIndices(i);

        % Add the number to the left of zero if it's not at the beginning
        if index > 1
            adjacentNumbers(end+1) = x(index - 1);
        end

        % Add the number to the right of zero if it's not at the end
        if index < length(x)
            adjacentNumbers(end+1) = x(index + 1);
        end
    end

    % Find the largest number among the adjacent numbers
    % If there are no adjacent numbers, return -inf
    if isempty(adjacentNumbers)
        y = [];
    else
        y = max(adjacentNumbers);
    end
end
```