CSCI 421 Project 2: The quick sort algorithm

Objectives

- Process singly-linked lists using recursive functions.

Problem description

(See exercise 6 on page 116.)

Write a quick sort function of type `int list -> int list`.

A brief summary of the quick sort algorithm: First pick an element and call it pivot. (The head of the list is an easy, if suboptimal, choice for the pivot.) Partition the rest of the list into two sublists, one with all the elements less than the pivot and another one with all the elements not less than the pivot. Recursively quick sort the sublists. Combine the two sublists and pivot into final sorted list.

Your solution must follow the style of the merge sort function on page 114 . Any helper functions you create must defined using `let`.

Your function should be saved in a file named ProjectTwo.sml. Turn in that file on Blackboard once you have working quick sort implementation.

Sample run

```
$ sml
Standard ML of New Jersey (64-bit) v110.99 [built: Thu Dec 24 11:47:23 2020]
- use "ProjectTwo.sml";
[opening ProjectTwo.sml]
[autoloading]
[library $SMLNJ-BASIS/basis.cm is stable]
[library $SMLNJ-BASIS/(basis.cm):basis-common.cm is stable]
[autoloading done]
val quickSort = fn : int list -> int list
val it = () : unit
- quickSort [3,1,4,8,9,5,7,6,2,0];
val it = [0,1,2,3,4,5,6,7,8,9] : int lis
- quickSort [~1,5,~3,9,11,2];
val it = [~3,~1,2,5,9,11] : int list
```

Grade breakdown

Criteria	Weight
Code is cleanly formatted & appropriately documented	20%
Correctness of `quickSort` function	30%
`quickSort` implements the quick sort algorithm (not some other sorting algorith)	20%
Helper functions are defined using `let`, ie are not top-level `funs`	30%

If your code doesn't compile without modifications, 20% will be deducted from your score.