C: FILESORT MANUAL

WRITTEN BY ARYAN SHAH AND ALLEN ZHANG

NAME

filesort - lexicographically sort tokens in text files

SYNOPSIS

./**filesort** [Option] [File]

DESCRIPTION

Write sorted tokens from a text file containing comma separated tokens to standard out.

If [Option] or [File] flags are missing, return fatal error.

Options:

-i
sort tokens using insertion sort

-q
sort tokens using quick sort

NAME

insertionsort - lexicographically sort tokens in linked lists

SYNOPSIS

```
int insertionSort(void* toSort,int (*comparator)(void*, void*));
```

DESCRIPTION

Sort nodes within type agnostic linked list toSort using insertion sort algorithm.

The insertion sort algorithm makes use of a function pointer to a comparator token responsible for comparing the ascii character values of the tokens stored within the nodes

RETURN VALUE

On insertion sort completion, insertionsort() returns a 1.

NAME

quicksort - lexicographically sort tokens in linked lists

SYNOPSIS

```
int quickSort(void* toSort,int (*comparator)(void*, void*));
```

DESCRIPTION

Sort nodes within type agnostic linked list toSort using quicksort algorithm.

The quicksort algorithm makes use of a function pointer to a comparator token responsible for comparing the ascii character values of the tokens stored within the nodes

RETURN VALUE

On quick sort completion, insertionsort() returns a 1.

NAME

dataComparator - compares the lexicographic value of two tokens

SYNOPSIS

```
int dataComparator(void* dataOne,void* dataTwo));
```

DESCRIPTION

Compare the lexicographic value of two type agnostic tokens.

Type agnostic pointers are casted as char* pointers to determine if tokens represent a string value or an integer value.

After determining the appropriate type of the data, a comparison of the two data values is made.

RETURN VALUE

dataComparator() returns a 1 if dataOne is lexicographically greater than or equal to datTwo.

dataComparator() returns a -1 if dataTwo is lexicographically greater than dataOne.