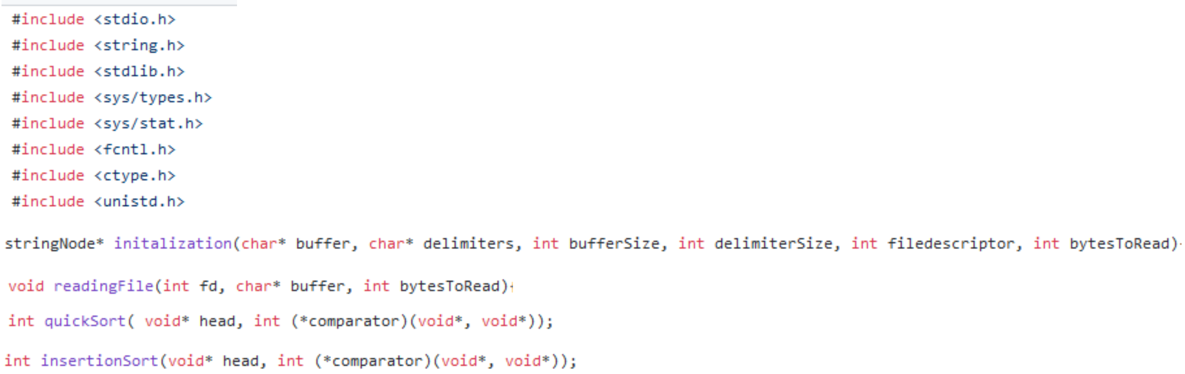
NAME

initialization, readingFile, quicksort, insertionsort

SYNOPSIS

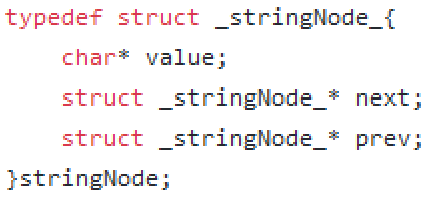


```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <ctype.h>
#include <unistd.h>

stringNode* initalization(char* buffer, char* delimiters, int bufferSize, int delimiterSize, int filedescriptor, int bytesToRead)

void readingFile(int fd, char* buffer, int bytesToRead)

int quickSort( void* head, int (*comparator)(void*, void*));

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

DESCRIPTION

The **initialization()** function returns a pointer to a **stringNode** structure, representing the head of the linked list of tokens. If there is an empty file or an error, it will return NULL. It creates a linked list of tokens from the given file and removes the necessary white space and sets the fileType of the file to 1 if is a number file and 0 if it is a string file.

The **stringNode** structure is defined as follows:



```
typedef struct _stringNode_{
    char* value;
    struct _stringNode_* next;
    struct _stringNode_* prev;
}stringNode;
```

The fields of the **stringNode** structure are as follows:

value is the string the token holds.

next is a pointer to another **stringNode** structure, usually after it in the linked list.

prev is a pointer to another **stringNode** structure, usually the node before it in the linked list.

The **readingFile()** function uses a file descriptor and opens it in read only mode and performs a blocking operation in which it attempts to either completely fill the buffer or until it hits EOF(end of file). Once done, the buffer given will either be completely filled or partially filled. Before filling the buffer, the buffer is initialized to NULL.

The **insertionSort()** function sorts the data through insertion sort and uses the comparator to compare the data.

The **quickSort()** function sorts the data in place and uses the comparator to compare the data.

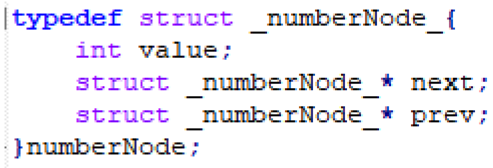
RETURN VALUE

On success, **initialization()** returns a **stringNode** structure which represents the head of the linked list of tokens.
On error or empty file, **initialization()** returns NULL.

For debugging purposes, **insertionSort** and **quickSort** both return 1 on success.

NOTES

If the file is a number file, the main() method will convert the stringNode structures into numberNode structures with the only difference in the field being the value in numberNode is now treated like an int instead of char*.



```
typedef struct _numberNode_{
    int value;
    struct _numberNode_* next;
    struct _numberNode_* prev;
}numberNode;
```

When using the binary executable, the appropriate file arguments is:

The first argument is: ./<executable binary>
The second argument is: -q for quicksort or -i for insertion sort
The third argument is the file to sort
For instance: ./<executable binary> -i unsorted.txt