#include <stdlib.h>

#include <stdio.h>

#include <string.h>

//#include "Strings.h"

typedef char\* String;

String getfc(FILE\* pFIn, String terminators, int n);

int charInString(String t, char c);

// a cover function for malloc()

// malloc and return memory for a string of stringsize characters

// return (char\*)NULL on failure

String mallocString(int stringsize)

{

    return (String)malloc(sizeof(char)\*(stringsize + 1));

}

// just a cover function for free()

void freeString(String s)

{

    free(s);

}

// create a duplicate string of s

// return it

// return (char\*)NULL on failure

// should call mallocString(), and then strcpy()

String duplicateString(String s)

{

    int strSize = sizeof(s);

    String strCopy = mallocString(strSize);

    if(strCopy == (char\*)NULL)

    {

        fprintf(stderr, "Failed to allocate memory\n");

    }

    strcpy(strCopy, s);

    return strCopy;

}

String\* duplicateStringList(String\* s, int n)

{

    String\* slCopy;

    // Allocate the list

    slCopy = (String\*)malloc(sizeof(String)\*n);

    if(slCopy == (String\*)NULL) return slCopy;

    // Allocate/duplicate the strings

    for(int i = 0; i < n; i++)

    {

        slCopy[i] = duplicateString(s[i]);

        if(slCopy[i] == (String)NULL)

        {

            // Bad stuff - clean up and return

            for(int j = 0; j < i; j++)

            {

                freeString(slCopy[j]);

            }

            free(slCopy);

            slCopy = (String\*)NULL;

            break;

        }

    }

    return slCopy;

}

char\* getString(char\* terminators, int n)

{

    char\* s;

    s = getfc(stdin, terminators, n);

    return s;

}

int compareString(const void \* arg1, const void \* arg2)

{

    const char \* ptr1 = (const char \*)arg1;

    const char \* ptr2 = (const char \*)arg2;

    if(strcmp(ptr1, ptr2)<0)

    {

        return -1;

    }

    if (strcmp(ptr1, ptr2)==0)

    {

        return 0;

    }

    return 1;

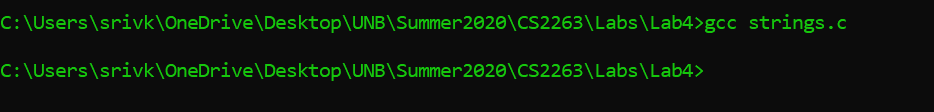
}

int main(int argc, char\* argv[])

{

}

//End



#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include "Strings.h"

int main(int argc, char\* argv[])

{

    char\* progName;

    progName = duplicateString(argv[0]);

    if(progName == (char\*)NULL)

    {

        fprintf(stderr, "Failed");

        return EXIT\_FAILURE;

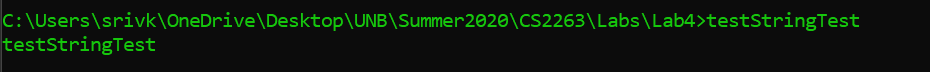
    }

    printf("%s\n", progName);

    freeString(progName);

    return EXIT\_SUCCESS;

}



#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include "Strings.h"

int main(int argc, char\* argv[])

{

    if(argc < 1)

    {

        int var = argc-1;

        char\*\* ch = duplicateStringList(argv+1, var);

        int num = 0;

        while(ch != (char\*)NULL && num < var)

        {

            printf("%s\n", ch[num]);

            num++;

        }

    }

    return 0;

}