CS2002 170025298 Tutor: Maywan Fayed
Date: 12th March 2019
Practical 2 – faro shuffle

1. Design

The whole program consists of two c. files, two header files (one is provided from practical folder) and one Makefile.

c. file includes faro_shuffle.c and functions.c, where functions.c is used to store different functions such as print whole content and shuffle decks, faro_shuffle.c is the main method and is used to check command line argument and get user_input from consoles.

faro_shuffle.c checks command line input and if the input does not fit to one of the RANKSUIT or NUMERICAL and the program will complain. Since “==” cannot be used to compare Strings, so I used strcmp() to compare strings. The the program will then check size and k using scanf(), actually there is a bug that I may need to check whether size is equal to the amount of decks and get whole decks by using fgets.

In main() function, the while loop will break when user input -1 as variable “Size”, or the while loop will read next decks.

Since for ranksuit there are 52 cards maximum and in Numerical part, the maximum size is UINT_MAX, so the program also checks overflow.

The functions.c implement program using linked list which is type defined as *decks, which consists of three elements inside of the structure: card value with type String (or to say char*), an index which is used to count the index of the node and one next with type *decks. The index will be used to check the pointer within shuffle() can reach to the half of the decks and will use index_reorder() to reorder linked list from 0 to size-1 again.

Test() is used to put a mark to the program and check the error during program execution to prevent the program crashed. Memory_check() will check the memory is enough or not, otherwise the program will return hints that tells people memory is not enough.

Function of “functions” are noted inside of the program.

After all operations has done, decks will be freed in main function and the program will receive next input until user input -1 in size or the program stopped because of invalid input.

2. Implementation:

Ranksuit : After reading variables – “k, size, deck” from user, the main method will calls ranksuit in functions.c to start the shuffle. Firstly the program will assigned the head to order_deck() which will split char* deck from a string to a linked list, then the program will create a list which malloc 256 bytes, which is enough to store 16 characters (the maximum k will be 2^16, it is almost impossible to give a large k since number of decks is limited.) and return the address of first element of the list. Then the program will read through the binary list and decide to do the in_shuffle or out_shuffle by identity I is 0 or 1 and call the index_reorder() to reorder the index from 0 to size-1. Then the program will print the content and loop the process again until k = -1, which is signed in k_binary as the final of the list.

3. Difficulty:(Backup means the file in backups where is the folder to store codes that not works)
functions.c : 175 : Since list have been freed in line 198, why cannot I free result in line 249 since if I free there again the compiler will complain that I cannot free it twice? What is the stack frame of that?

Functiosns.c : 90 : Since in free_all(decks* head), head is a local variable of free_all function, will head be freed with this function?

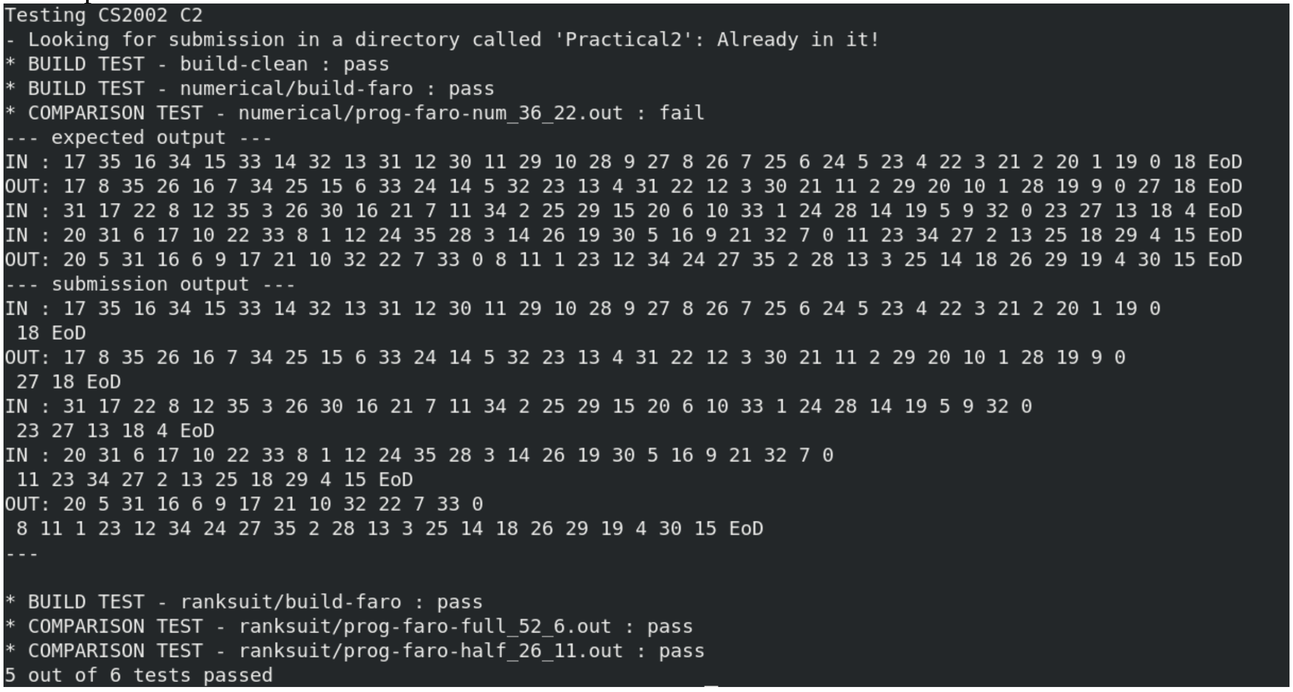
functions.c : 183 : Is there an easier way to read binary list from last to first (from left to right)?

Backups – functions_ptr.c : How can I use pointers to store variables as lists?

Backups – faro_shuffle.c : line 60 : How can I use read char to tackle the input value without using scanf(); since the while loop does not work even I press "ENTER"

(Marked)Backups – functions_1.c : Line 174 : while making it it shows a strange index with strange cards as what I have sent in the e-mail, why this happens? How can I tackle them?

4. Output:



```
Testing CS2002 C2
- Looking for submission in a directory called 'Practical2': Already in it!
* BUILD TEST - build-clean : pass
* BUILD TEST - numerical/build-faro : pass
* COMPARISON TEST - numerical/prog-faro-num_36_22.out : fail
--- expected output ---
IN : 17 35 16 34 15 33 14 32 13 31 12 30 11 29 10 28 9 27 8 26 7 25 6 24 5 23 4 22 3 21 2 20 1 19 0 18 EoD
OUT: 17 8 35 26 16 7 34 25 15 6 33 24 14 5 32 23 13 4 31 22 12 3 30 21 11 2 29 20 10 1 28 19 9 0 27 18 EoD
IN : 31 17 22 8 12 35 3 26 30 16 21 7 11 34 2 25 29 15 20 6 10 33 1 24 28 14 19 5 9 32 0 23 27 13 18 4 EoD
IN : 20 31 6 17 10 22 33 8 1 12 24 35 28 3 14 26 19 30 5 16 9 21 32 7 0 11 23 34 27 2 13 25 18 29 4 15 EoD
OUT: 20 5 31 16 6 9 17 21 10 32 22 7 33 0 8 11 1 23 12 34 24 27 35 2 28 13 3 25 14 18 26 29 19 4 30 15 EoD
--- submission output ---
IN : 17 35 16 34 15 33 14 32 13 31 12 30 11 29 10 28 9 27 8 26 7 25 6 24 5 23 4 22 3 21 2 20 1 19 0
 18 EoD
OUT: 17 8 35 26 16 7 34 25 15 6 33 24 14 5 32 23 13 4 31 22 12 3 30 21 11 2 29 20 10 1 28 19 9 0
 27 18 EoD
IN : 31 17 22 8 12 35 3 26 30 16 21 7 11 34 2 25 29 15 20 6 10 33 1 24 28 14 19 5 9 32 0
 23 27 13 18 4 EoD
IN : 20 31 6 17 10 22 33 8 1 12 24 35 28 3 14 26 19 30 5 16 9 21 32 7 0
 11 23 34 27 2 13 25 18 29 4 15 EoD
OUT: 20 5 31 16 6 9 17 21 10 32 22 7 33 0
 8 11 1 23 12 34 24 27 35 2 28 13 3 25 14 18 26 29 19 4 30 15 EoD
---

* BUILD TEST - ranksuit/build-faro : pass
* COMPARISON TEST - ranksuit/prog-faro-full_52_6.out : pass
* COMPARISON TEST - ranksuit/prog-faro-half_26_11.out : pass
5 out of 6 tests passed
```

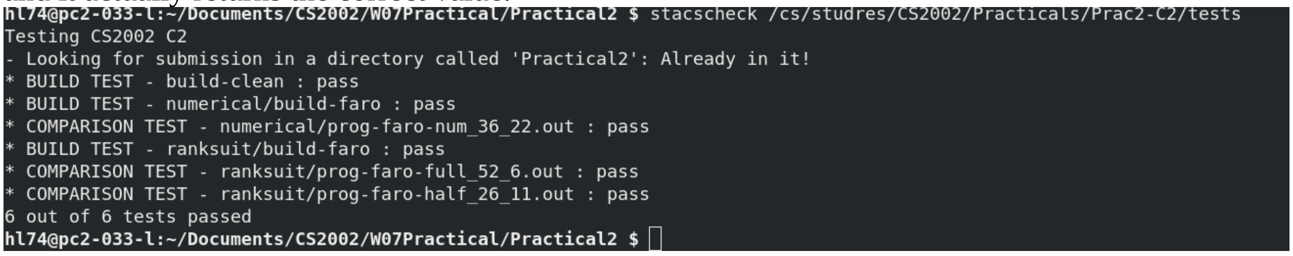
5. Testing:
Several testing available in the program:
functions.c:
Line 39, line 65 tests the memory has allocated so that the node can accept the value of index.
Line 83 is used to test index is successfully reordered.
Line 256 and 275 tests that variable "i" is as same as its algorithm prints.

The reason why in out_shuffle I put index of ptr1 and ptr2 is because index can be used to check the shuffle is right.

Tests in out_shuffle can be seen in backed_up which aims at finding the problem by checking all values.

All function test() are used to stop the program to finish the execution in order to prevent execution failed that shuts the program without printing any messages.

Extension:
Numerical: Since one thing that quite confused is that there always be one \n after 0 since that the result always goes wrong. So I set the end of the line from '\n' to ' 'such that \n will not being read and it actually returns the correct value.



```
hl74@pc2-033-l:~/Documents/CS2002/W07Practical/Practical2 $ stacscheck /cs/studres/CS2002/Practicals/Prac2-C2/tests
Testing CS2002 C2
- Looking for submission in a directory called 'Practical2': Already in it!
* BUILD TEST - build-clean : pass
* BUILD TEST - numerical/build-faro : pass
* COMPARISON TEST - numerical/prog-faro-num_36_22.out : pass
* BUILD TEST - ranksuit/build-faro : pass
* COMPARISON TEST - ranksuit/prog-faro-full_52_6.out : pass
* COMPARISON TEST - ranksuit/prog-faro-half_26_11.out : pass
6 out of 6 tests passed
hl74@pc2-033-l:~/Documents/CS2002/W07Practical/Practical2 $
```

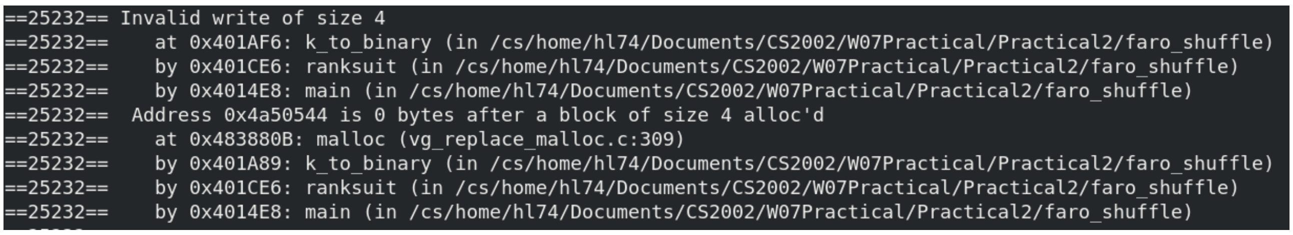
The implementation of this one is the same as what it has done in RANKSUIT, but since numerical only accept numbers, so check numerical is added to the function to check all values in linked lists are digits.

Extension 2:
Since I am not sure whether my correction will cause a good or bad result, so any possible correction will only be changed in extension file.

Valgrind is a debugger tool which can be used to check memory leak. By typing:

valgrind --leak-check=yes ./faro_shuffle RANKSUIT <sample_input.txt

The output of using this debugger shows that



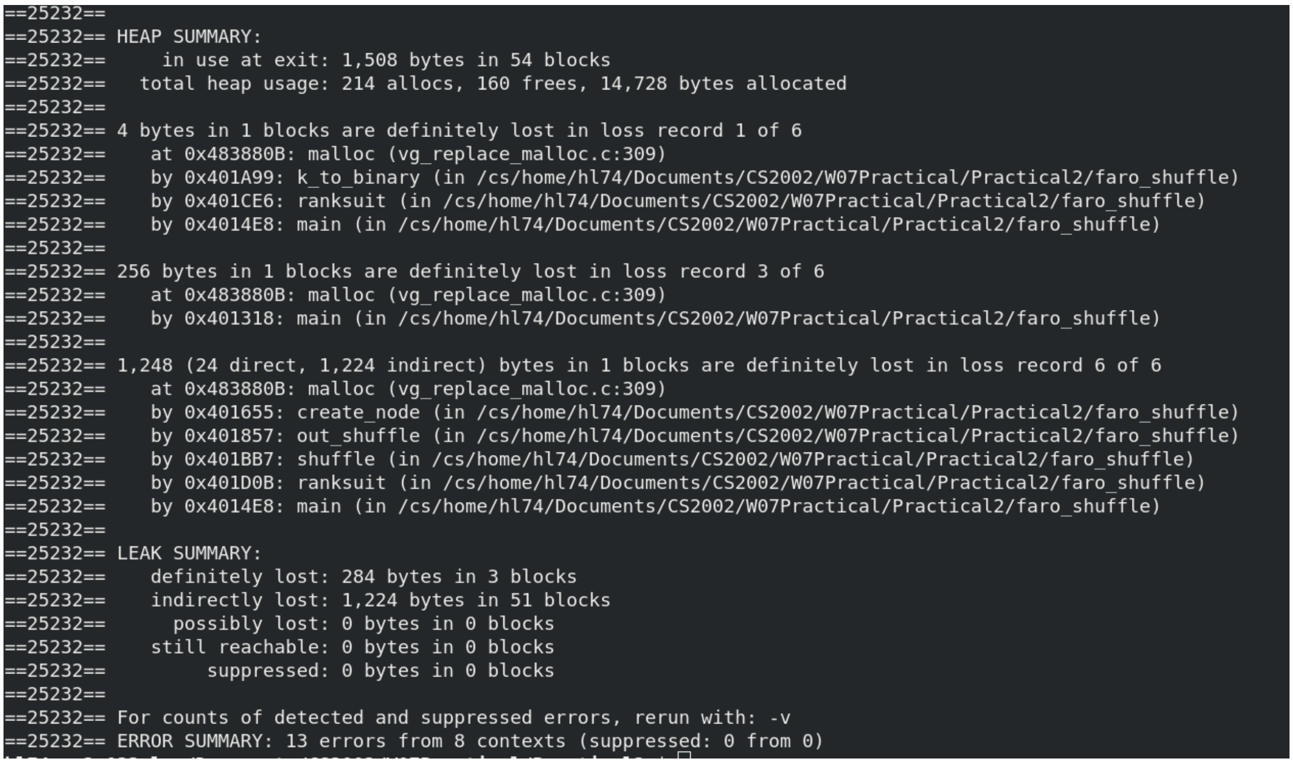
```
==25232== Invalid write of size 4
==25232==    at 0x401AF6: k_to_binary (in /cs/home/hl74/Documents/CS2002/W07Practical/Practical2/faro_shuffle)
==25232==    by 0x401CE6: ranksuit (in /cs/home/hl74/Documents/CS2002/W07Practical/Practical2/faro_shuffle)
==25232==    by 0x4014E8: main (in /cs/home/hl74/Documents/CS2002/W07Practical/Practical2/faro_shuffle)
==25232==  Address 0x4a50544 is 0 bytes after a block of size 4 alloc'd
==25232==    at 0x483880B: malloc (vg_replace_malloc.c:309)
==25232==    by 0x401A89: k_to_binary (in /cs/home/hl74/Documents/CS2002/W07Practical/Practical2/faro_shuffle)
==25232==    by 0x401CE6: ranksuit (in /cs/home/hl74/Documents/CS2002/W07Practical/Practical2/faro_shuffle)
==25232==    by 0x4014E8: main (in /cs/home/hl74/Documents/CS2002/W07Practical/Practical2/faro_shuffle)
```

In this part, function k_to_binay() caused a memory leak, which is caused by deck* list is freed but the result deos not, however this confused me as what I have described in difficulty above.

Since the overall memory leak is caused by the same issue and this is the whole memory leak issue that my program has made.

To try to fix the code, the only way should be free the result after use in Ranksuit, however if I assigned k_list to k_to_binary result and free it, it will return that it was double freed.

As the result, the total memory leak of my program shows as follows:



```
==25232==
==25232== HEAP SUMMARY:
==25232==     in use at exit: 1,508 bytes in 54 blocks
==25232==   total heap usage: 214 allocs, 160 frees, 14,728 bytes allocated
==25232==
==25232== 4 bytes in 1 blocks are definitely lost in loss record 1 of 6
==25232==    at 0x483880B: malloc (vg_replace_malloc.c:309)
==25232==    by 0x401A99: k_to_binary (in /cs/home/hl74/Documents/CS2002/W07Practical/Practical2/faro_shuffle)
==25232==    by 0x401CE6: ranksuit (in /cs/home/hl74/Documents/CS2002/W07Practical/Practical2/faro_shuffle)
==25232==    by 0x4014E8: main (in /cs/home/hl74/Documents/CS2002/W07Practical/Practical2/faro_shuffle)
==25232==
==25232== 256 bytes in 1 blocks are definitely lost in loss record 3 of 6
==25232==    at 0x483880B: malloc (vg_replace_malloc.c:309)
==25232==    by 0x401318: main (in /cs/home/hl74/Documents/CS2002/W07Practical/Practical2/faro_shuffle)
==25232==
==25232== 1,248 (24 direct, 1,224 indirect) bytes in 1 blocks are definitely lost in loss record 6 of 6
==25232==    at 0x483880B: malloc (vg_replace_malloc.c:309)
==25232==    by 0x401655: create_node (in /cs/home/hl74/Documents/CS2002/W07Practical/Practical2/faro_shuffle)
==25232==    by 0x401857: out_shuffle (in /cs/home/hl74/Documents/CS2002/W07Practical/Practical2/faro_shuffle)
==25232==    by 0x401BB7: shuffle (in /cs/home/hl74/Documents/CS2002/W07Practical/Practical2/faro_shuffle)
==25232==    by 0x401D0B: ranksuit (in /cs/home/hl74/Documents/CS2002/W07Practical/Practical2/faro_shuffle)
==25232==    by 0x4014E8: main (in /cs/home/hl74/Documents/CS2002/W07Practical/Practical2/faro_shuffle)
==25232==
==25232== LEAK SUMMARY:
==25232==    definitely lost: 284 bytes in 3 blocks
==25232==    indirectly lost: 1,224 bytes in 51 blocks
==25232==      possibly lost: 0 bytes in 0 blocks
==25232==    still reachable: 0 bytes in 0 blocks
==25232==         suppressed: 0 bytes in 0 blocks
==25232==
==25232== For counts of detected and suppressed errors, rerun with: -v
==25232== ERROR SUMMARY: 13 errors from 8 contexts (suppressed: 0 from 0)
```

Extension 3:
Since I have tried a lot to try to print UTF-8 code to the console, however it never works that the console never prints the code as I wish.

However I can describe some of my implementation, decks will be ordered and then transfer to UTF-8 code using encoding which can change H.S.D.C to their different UTF-8 characters.

However encoding was ignored by unknown reason even the output and test did not changed at all evenif I use wprintf().