Hacettepe University Computer Science and Engineering Department BBM203 ASSIGNMENT 2

November 12, 2017

Name and Surname :FURKAN KARAKÖKÇEK
Identity Number : 21328155
Course : BBM203
Subject : Stack,Queue,Dynamic Memory Allocation
Data Due : 12.11.2017
e-mail : furkankarakokcek@gmail.com
Advisors :R.A Gültekin Işık, Dr. Burcu Can, Dr. Sevil Şen, Dr. Adnan Özsoy

1 Introduction

In this experiment we are responsible from learning stack and queue structures of C programming language.This structures are used on basic client-server simulation.The simulation has at least client and only one server.Clients have stack and queue for process or interrupt.Server also has two structures and execute the process or interrupt for it's current situation.At this simulation,stack structure is more priority than queue structure.For example,when we used send command or execute command,algorithm firstly checks the stack structure if it is not available than uses queue structure.If there are some errors in the executing or sending part, algorithm will save the type of the error to log file of each client and server.

2 Software Using Documentation

2.1 Software Usage

This software based on C programming language.I compiled on dev machine using this command "gcc hw2.c -o output". Software takes two command line argument.They are input file of this software.First argument contains the client/server's stack and queue size.Last line is about server and the other lines about clients.Second argument is about commands.
First lines of the input files are line number of this files.
Software runs this functions sequentially;

ReadingTotalLine:Reads first line of the files.

ReadingLines:Reads the all information about the stack and queue sizes.

Commands:Reads all commands and execute them according command.

Write:Creates a output file which is third argument line command and write all log history to this file.

Free:This function frees the allocated struct array of clients and server.
Controller functions of this software;

IsStackFull:Checks the stack availability and returns 1 if stack full,otherwise returns 0.

IsStackEmpty:Checks the stack availability and returns 1 if stack empty,otherwise returns 1.

Push:Add single character to stack which second parameter of this function and incerements the top of stack.

Pop:Decrements the top stack and doesn't really remove the elements.

GetElementStack:Returns the current top element of the stack.

IsQueueFull:Checks availability of queue and returns 1 if queue is full,otherwise returns 0.

IsQueueEmpty:Checks availability of queue and returns 1if queue is empty,otherwise returns 0.

Enqueue:Add new element to rear of queue and incerements rear by 1.Second parameter of function is new element.

Dequeue:Incerements the front of the queue and doesn't really remove the element.

3 Software Design Notes

3.1 Desctiption of the program

3.1.1 Problem

My point of view,problem checking all possibilities of commands.Because,if the software doesn't the check all possibilities,at the output file there will be so many wrong process or interrupt name and error numbers.The another problem is allocating space for array of structure and content of the structure which are stack and queue.

3.1.2 Solution

My solutions are controlling stacks and queue availability before adding or removing operations.This was solved the many steps of problems.For allocating problem,I used malloc function.

3.1.3 Algorithm

1.Read first line argument
1.1.Find number of clients and server.
1.2. Allocate array of struct with this number.
1.3.Read the each line using fscanf function.
2.For all elements in the array of struct.
2.1. Initialize top value to -1.
2.2. Initialize front value to -1.
2.3. Initialize rear value to -1.
2.4. Initialize logindex value to -1.
2.5 Allocate stack and queue array.
3.Read second line argument
3.1.For each command in second input file
3.2.Control the first character.
3.3.If command begins with 'A',add third character to given specific client queue which is second character.

3.4.If command begins with 'I',add third character to given specific client/server stack which is second character.
3.5.If command begins with 'S',check available element of specific client which is again second character.Then send this element to server queue.
3.6.If command begins with 'O',check available element of server and remove it from server.
3.7.If there is an error at during control step,add error number to loghistory.
3.8.Add current command operation to loghistory with specific process or interrupt name.
4.Write all loghistory arrays to output file which is third argument line command.
5. Free allocated spaces.

4 Software Testing Notes

4.1 Bugs and Software Reliability

Bugs of this software are;
1.If you command "A SERVERIndex x",this software will add server queue 'x' character.
2.If you command "I ClientIndex x",If ClientIndex is greater than from number of clients,there won't be error message.
3.I initialized
the loghistory array with 500.This may causes the error with huge command file.

REFERENCES

www.programiz.com/c-programming/c-dynamic-memory-allocation
www.cprogramming.com/tutorial/cfileio.html
www.cs.bu.edu/teaching/c/stack/array/
stackoverflow.com/questions/24753342/
c-how-to-free-a-struct-array-cell-completely
www.programiz.com/dsa/circular-queue