Design of Assignment3

Simulation of memory

Angel Jimenez

Tyler James McConnell

Miguel Velazquez

May 2, 2016

Version 1.0

**Table of Contents**

Introduction.......................................................................2

Design Overview...............................................................2

System Architecture..........................................................3

Interaction.........................................................................4

**Introduction**

The purpose of this document is to describe the implementation of having to write a simulation that explores the effects of limited memory and memory policies. The simulator will read policy information and the characteristics of the workload from the input files and then will simulate the execution of the processes as well as decisions of Memory Manager. The simulator will generate an output file with the trace of important events, as well as the memory map and the input queue status after each event. AT the end of the simulation, the program will also print the average turnaround time per process.

**Design Overview**

**Description**

Main.cpp

Implementation of the whole program

1. Creates a class Process

2. Creates a structure memory block

3. Opens file

4. Sets memory size and page size to 0

5. Creates vectors of process queue, memory block, events, pages, and ready queue

6. User enters memory size and page size

7. Switch statement comes into play depending on choice made by user

8. Function read file gets the info in process queue adn events and build memory map gets info form memory block, page size and memory size.

9. Then it says how long it has been in memory in milliseconds

10. It releases the finished processes and says which one it’s removing

11. Compares memory size and memory block to time and if equal it removes it

12. Else it compares if end of processes is equal to false and process time start is equal to time and if there is available memory it adds process to ready queue first.

13. increases e\_time.

14. checks if size is greater than 0 and if the ready back start time is equal to time or if the ready back time end is equal to time then checks to see if memory is available if so it adds process to memory

15. Else it increases start time and time end

16. Than if memory back time end is equal to time it removes process

17. Will print out the PID number, time start and time end and life time and the total memory

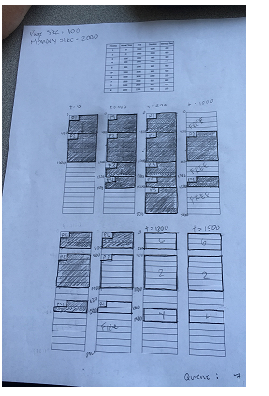
18. Repeat steps 7-18 until finished

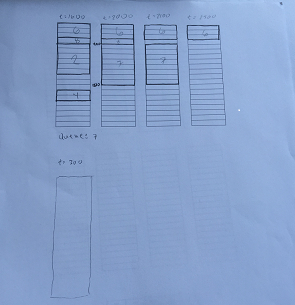
19. Prints out the average turnaround time

20. Program exits

**System Architecture**

Below is diagram of how different processes would look in memory depending when they arrived. Also all had different time duration and required memory.

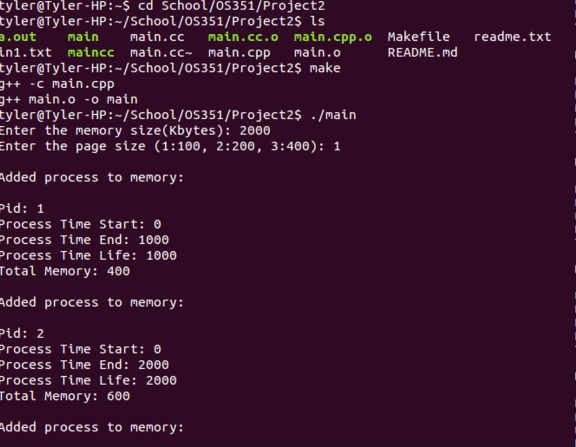




**Interaction**

Process\_memory uses the command line interface to interact with the user. After using make to compile the files into executable on a UNIX based/Linux operating system, the user can run the main file

Use the ls statement then the make statement. After that it will say to enter the memory size and enter it and then page size and enter it and will say added to process to memory



Then at the very end it will give the average turnaround time

