P536: ADVANCED OPERATING SYSTEMS

ASSIGNMENT 6: REPORT

TEAM: ROHIT PATIL, SAMEEKSHA VAITY

Files created for this Assignment:

- shell -> xsh_newmem.c
- include -> newmem.h
- system -> newgetstk.c, newcreate.c, killme.c
- apps -> testprocess.c

9.1 Write a function that walks the list of free memory blocks and prints a line with the address and length of each block.

Solution:

Shell command to execute Question no 1(9.1) – newmem –f

Created a new function –"printfreelist()" which prints address and the length of the free memory block.

9.3 Replace the low-level memory management functions with a set of functions that allocate heap and stack memory permanently (i.e., without providing a mechanism to return storage to a free list). How do the sizes of the new allocation routines compare to the sizes of getstk and getmem?

Solution:

Shell command to execute Question no 2(9.3) – newmem –s

This shell command creates a process named "test process" using -"newcreate" system call.

newcreate system call uses "newgetstk" to allocate stack space for the process.

This process exits by a call to "killme" system call which kills the process without freeing the stack memory and heap memory.

1. *getstk vs newgetstk*

- getstk implements "last fit" allocation which is it walks through the entire free memory list and allocates the last available block which fits the requested process stack size.
- Whereas, newgetstk allocates stack space to the process using first fit allocation method i.e. allocates the first available free block starting from the lowest address which fits the requested process stack size.

2. *getmem*

- For allocating heap storage we have to explicitly call getmem and deallocation of the heap storage is done by an explixit call to freemem.
- In our implementation we allocate the heap storage using getmem but there is no explicit call made to freemem so the heap storage is never deallocated.

If we chose to free the memory (heap and stack) which is allocated using our implementation routines it will lead to internal fragmentation of the free memory list creating holes in the memory block.

9.7 Many embedded systems go through a prototype stage, in which the system is built on a general platform, and a final stage, in which minimal hardware is designed for the system.In terms of memory management, one question concerns the size of the stack needed by each process. Modify the code to allow the system to measure the maximum stack space used by a process and report the maximum stack size when the process exits.

Solution:

Shell command to execute Question no 3(9.7) – newmem –t

Created a process called “testprocess” using the “newcreate” system call.

Initialized the entire stack space of this process with a default value.

Process is then resumed to use the same stack space.

When the process exits by a call to the “killme” system call, we walk through the entire stack space reporting the maximum used space by the process. This is done by counting the number of non-default values found in the stack space.