ASSIGNMENT 1

Seth Lunders
Professor Bolden
CS 240
March 12

Program Design

The programs were all very simple to write. Here is the pseudocode for each of them:

printPID.c

```
int i;
for i from 0 to 10 {
    print Name, PID, and Iteration count
}
```

infLoopCounter.c

```
int i;
while(1){
    i++;
}
```

infLoopFileWrite.c

```
FILE* fileOut = file name from argv
while(1){
    open fileOut
    write character to fileOut
    close fileOut
}
```

Programming Log

- Mar 10 | Planned & wrote the programs
  - Looked over assignment (10min)
  - Wrote printPID.c code (25min)
  - Wrote infLoopCounter.c and infLoopFileWrite.c (45min)
- Mar 12 | Ran test with the programs
  - Tested each program with top
  - Wrote this report

Source Code

printPID.c

```
/* a2.c
 * CS 240.Bolden.........Seth Lunders
 * 3/18/2021.............lund4272@vandals.uidaho.edu
 * This program prints out its process ID
 * gcc version 10.2.0
 *  ------------------------------------------------
 */

#include <stdio.h>
#include <unistd.h>
// ---------------
// Main Program
// ---------------
int main(int argc, char* argv[])
{
    int i;
    for (i = 0; i < 10; i++)
    {
        fprintf(stderr, "Name: %s PID: %i Iteration: %i\n",
argv[0],getpid(),i);
        sleep(1);
    }
}
```

infLoopCounter.c

```c
#include <stdio.h>
// ---------------
// Main Program
// ---------------
int main()
{
    // Create an integer
    int i = 0;

    // Print the process name and PID.
    fprintf(stderr, "Name: %s PID: %i\n", argv[0], getpid());

    while(1){
        i++;
    }
}
```

infLoopFileWrite.c

```c
#include <stdio.h>
#include <unistd.h>
// ---------------
// Main Program
// ---------------
int main(int argc, char *argv[])
{
    // Create file pointer
    if(argv[1] == NULL){
        printf("Please specify output file\n");
        return 1;
    }
    FILE *fileOut;

    // Print the process name and PID.
    fprintf(stderr, "Name: %s PID: %i\n", argv[0], getpid());

    // Infinite loop that writes 'c' to the input file
```

```
    while (1)
    {
        // Set pointer to the input file. Creates file if it does not exist.
        fileOut = fopen(argv[1], "w");
        // Write to the file
        fputc('c', fileOut);
        // Close the file
        fclose(fileOut);
    }
}
```

Output

printPID.c

```
Script started on 2021-03-12 17:45:17-08:00 [TERM="xterm-256color"
TTY="/dev/pts/3" COLUMNS="197" LINES="18"]
]0;sethlunders@pop-os: /media/data/College/CS240/a1[[01;32msethlunders@pop-
os[00m:[01;34m/media/data/College/CS240/a1[00m$ ./printPIDLoop
Name: ./printPIDLoop PID: 39947 Iteration: 0
Name: ./printPIDLoop PID: 39947 Iteration: 1
Name: ./printPIDLoop PID: 39947 Iteration: 2
Name: ./printPIDLoop PID: 39947 Iteration: 3
Name: ./printPIDLoop PID: 39947 Iteration: 4
^Z
[1]+  Stopped                 ./printPIDLoop
]0;sethlunders@pop-os: /media/data/College/CS240/a1[[01;32msethlunders@pop-
os[00m:[01;34m/media/data/College/CS240/a1[00m$ fg %1
./printPIDLoop
Name: ./printPIDLoop PID: 39947 Iteration: 5
Name: ./printPIDLoop PID: 39947 Iteration: 6
Name: ./printPIDLoop PID: 39947 Iteration: 7
Name: ./printPIDLoop PID: 39947 Iteration: 8
Name: ./printPIDLoop PID: 39947 Iteration: 9
^Z
[1]+  Stopped                 ./printPIDLoop
]0;sethlunders@pop-os: /media/data/College/CS240/a1[[01;32msethlunders@pop-
```

```
os[00m:[01;34m/media/data/College/CS240/a1[00m$ exit
exit
There are stopped jobs.
]0;sethlunders@pop-os: /media/data/College/CS240/a1[01;32msethlunders@pop-
os[00m:[01;34m/media/data/College/CS240/a1[00m$ exit
exit

Script done on 2021-03-12 17:45:50-08:00 [COMMAND_EXIT_CODE="1"]
```

infLoopCounter.c & infLoopFileWrite.c

No output other than looking at the CPU and RAM usage, which I go over in the results section.

Results & Observations

printPID.c

The first program worked well, it outputs its name, PID, and the iteration every 1 second.

infLoopCounter.c

The first cpu-intensive program infinitely increments an integer. When I run it in the background, then check it with 'top', it is using 100% of the available cpu. The memory usage is listed as 72kb in the System Monitor app.

infLoopFileWrite.c

The second cpu intensive program opens a file, writes 'c' to it, closes it, then repeats infinitely. Its CPU usage hovers around 37%. The memory usage is also listed as 72kb.

Both CPU Intensive Programs at once

When running both programs, I was surprised by the CPU usage. The counter loop still hovered around 99%, while the file write also stayed relatively high, at 27%. This made me remember that my laptop has 8 cores, so I checked the System Monitor application which lists the cpu usage of each individual core. The cores hovered around 50% usage overall, but some jumped up to 100%. This may explain why the usage added up to over 100% in top.
I also checked the Memory usage, which was listed as 72kb for both programs.

It makes sense, but it's interesting to me that a program can use so little memory while maxing out the CPU.
