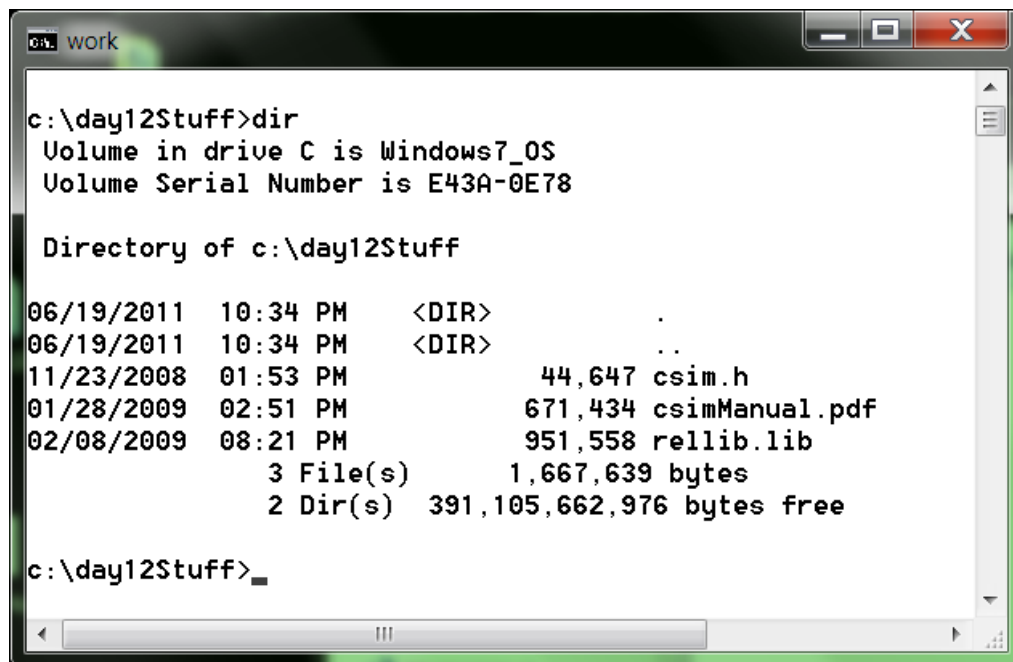


Objectives:

- 1) Get CSIM installed and be able to build a model
- 2) Understand what the CSIM manual contains
- 2) Build and run simple M/M/1 model
- 3) Understand CSIM process model
- 4) Review command line parameters and file I/O

Installing CSIM (for building with VisualExpress 2008 cl)

- Really nothing to install, just need three key files in your working directory



```
work
c:\day12Stuff>dir
Volume in drive C is Windows7_OS
Volume Serial Number is E43A-0E78

Directory of c:\day12Stuff

06/19/2011  10:34 PM    <DIR>          .
06/19/2011  10:34 PM    <DIR>          ..
11/23/2008  01:53 PM                44,647 csim.h
01/28/2009  02:51 PM            671,434 csimManual.pdf
02/08/2009  08:21 PM            951,558 rellib.lib
              3 File(s)        1,667,639 bytes
              2 Dir(s)  391,105,662,976 bytes free

c:\day12Stuff>_
```

- Where to find these files?

`\Windows\vs2008Express_student\c\lib`

- To get started let's download `mm1_csim.c` and `process_csim.c` from class handouts page and build and run them

- So, now we have

```

c:\day12Stuff>dir
Volume in drive C is Windows7_OS
Volume Serial Number is E43A-0E78

Directory of c:\day12Stuff

06/19/2011  10:51 PM    <DIR>          .
06/19/2011  10:51 PM    <DIR>          ..
11/23/2008  01:53 PM             44,647 csim.h
01/28/2009  02:51 PM           671,434 csimManual.pdf
01/27/2004  04:44 AM             8,695 mm1_csim.c
06/09/2009  09:58 AM             5,381 process_csim.c
02/08/2009  08:21 PM           951,558 rellib.lib
                5 File(s)      1,681,715 bytes
                2 Dir(s)  391,092,224,000 bytes free

c:\day12Stuff>

```

- Now we will build and run the M/M/1 model

```

c:\work>cl mm1_csim.c rellib.lib
Microsoft (R) 32-bit C/C++ Optimizing Compiler Version 15.00.21022.08 for 80x
Copyright (C) Microsoft Corporation. All rights reserved.

mm1_csim.c
Microsoft (R) Incremental Linker Version 9.00.21022.08
Copyright (C) Microsoft Corporation. All rights reserved.

/out:mm1_csim.exe
mm1_csim.obj
rellib.lib

c:\work>mm1_csim
*** BEGIN SIMULATION ***
=====
==      *** CSIM M/M/1 queueing system simulation ***      ==
=====
= Lambda   = 1.000 cust/sec
= Mu       = 3.000 cust/sec
=====
= Total CPU time   = 0.078 sec
= Total sim time  = 100000.000 sec
= Total completions = 100173 cust
=====
= >>> Simulation results
-----
= Utilization      = 33.224 %
= Mean num in system = 0.499 cust
= Mean response time = 0.498 sec
= Mean service time = 0.332 sec
= Mean throughput  = 1.002 cust/sec
-----
= >>> Theoretical results
-----
= Utilization      = 33.333 %
= Mean num in system = 0.500 cust
= Mean response time = 0.500 sec
= Mean service time = 0.333 sec
= Mean throughput  = 1.000 cust/sec
=====
*** END SIMULATION ***

c:\work>

```

- Are the results reasonable? Let's see...
 - We know the $L = \rho / (1 - \rho)$ for an M/M/1 and that $\rho = \lambda/\mu$.
 - So... $\rho = 1/3$ and $L = (1/3) / (1 - (1/3)) = 0.5$
- We will come back to `mm1_csim.c` in a moment
- Let's build and run `process_csim.c`...

```

VC++ cmd
c:\work>cl process_csim.c rellib.lib
Microsoft (R) 32-bit C/C++ Optimizing Compiler Version 15.00.21022.08 for 80x86
Copyright (C) Microsoft Corporation. All rights reserved.

process_csim.c
Microsoft (R) Incremental Linker Version 9.00.21022.08
Copyright (C) Microsoft Corporation. All rights reserved.

/out:process_csim.exe
process_csim.obj
rellib.lib

c:\work>process_csim
Begin at time = 0.000000
Hello from process #1 at time = 0.000000
Hello from process #1 at time = 1.000000
Hello from process #2 at time = 1.000000
Hello from process #1 at time = 2.000000
Hello from process #2 at time = 2.500000
Hello from process #1 at time = 3.000000
Hello from process #2 at time = 4.000000
Hello from process #1 at time = 4.000000
Hello from process #1 at time = 5.000000
Hello from process #2 at time = 5.500000
Hello from process #1 at time = 6.000000
Hello from process #2 at time = 7.000000
Hello from process #1 at time = 7.000000
Hello from process #1 at time = 8.000000
Hello from process #2 at time = 8.500000
Hello from process #1 at time = 9.000000
Hello from process #2 at time = 10.000000
Hello from process #2 at time = 11.500000
Hello from process #2 at time = 13.000000
Hello from process #2 at time = 14.500000
End at time = 101.000000

c:\work>

```

- Let's understand what is going on here...
 - CSIM processes (sort of like a thread... run independently)
 - Use of hold() to advance simulation time

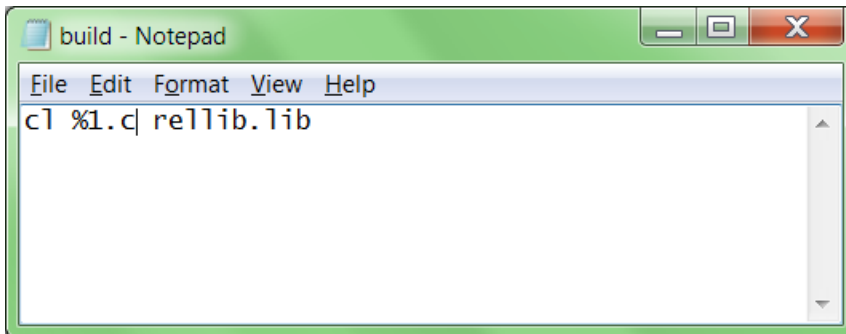
```
//=====
//==  CSIM process #1                               ==
//=====
void process1()
{
    int i;          // Loop counter

    // Create the process
    create("process1");

    // Loop 10 times holding between each iteration
    for (i=0; i<10; i++)
    {
        printf("Hello from process #1 at time = %f \n", clock);
        hold(1.0);
    }
}
```

Walk-through entire program focusing on the CSIM create() and hold() functions as found in main() (note... called sim()) and in the process1() and process2()

- Getting tired of typing cl ____.c rellib.lib so create a batch file...



- From now on we just type build ____ where ____ is the CSIM model file name (the .c file)
- Let's go back to mm1_csim.c and run it for different values of lambda, mu, and SIM_TIME
 - Demo what happens as SIM_TIME is decreased (faster run time, but results are increasingly off) and increased (longer run time, but results are better)

- All this edit/build/run/edit/build/run is a pain so let's add command line arguments

```

void sim(int argc, char *argv[])
{
    double    lambda;        // Mean arrival rate (cust/sec)
    double    mu;           // Mean service rate (cust/sec)
    double    rho;          // Theoretical utilization

    // Create the simulation
    create("sim");

    // CSIM initializations
    Server = facility("Server");

    // Output usage
    if (argc != 3)
    {
        printf("Usage: 'mml_csim lambda mu' \n");
        exit(1);
    }

    // Parameter initializations
    lambda = atof(argv[1]);
    mu = atof(argv[2]);
    assert(lambda < mu);
}

```

- And, now...

```

c:\day12Stuff>mml_csim 0.8 1.0
*** BEGIN SIMULATION ***
=====
==      *** CSIM M/M/1 queueing system simulation ***      ==
=====
= Lambda    = 0.800 cust/sec
= Mu        = 1.000 cust/sec
=====
= Total CPU time    = 0.109 sec
= Total sim time   = 100000.000 sec
= Total completions = 80181 cust
=====
= >>> Simulation results
-----
= Utilization      = 79.931 %
= Mean num in system = 3.969 cust
= Mean response time = 4.950 sec
= Mean service time = 0.997 sec
= Mean throughput  = 0.802 cust/sec
-----
= >>> Theoretical results
-----
= Utilization      = 80.000 %
= Mean num in system = 4.000 cust
= Mean response time = 5.000 sec
= Mean service time = 1.000 sec
= Mean throughput  = 0.800 cust/sec
=====
*** END SIMULATION ***
c:\day12Stuff>

```

- File I/O is important for assignment #5 and the project, so let's review...

```
// fio.c
#include <stdio.h>
#include <stdlib.h>

void main()
{
    FILE    *fp;
    char    inString[80];
    double  val;

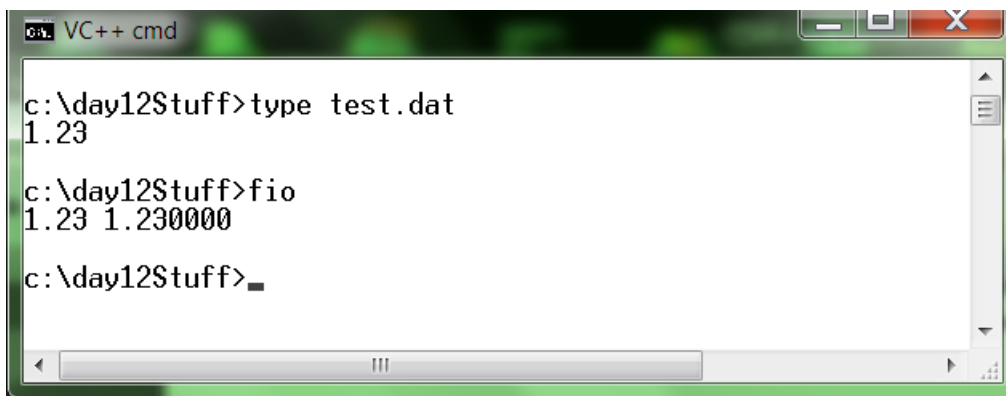
    fp = fopen("test.dat", "r");
    if (fp == NULL)
    {
        printf("*** ERROR - file not found \n");
        exit(1);
    }

    fscanf(fp, "%s", inString);
    val = atof(inString);
    printf("%s %f \n", inString, val);

    fclose(fp);
}
```

Create/edit test.dat with notepad and note how notepad sometimes sticks a .txt on the end of file names.

- And, we execute (after a cl build)...



```
VC++ cmd
c:\day12Stuff>type test.dat
1.23
c:\day12Stuff>fio
1.23 1.230000
c:\day12Stuff>_
```

- Finally... review assignment #5 and write on the board what needs to be done (but, don't write code on the board – just English description/hints).