Operating Systems (CMPS 272)

Lecture 1: Overview

Mohamad Jaber

American University of Beirut
-
Faculty of Arts & Sciences - Department of Computer Science

August 30, 2017

Outline

1. Administrative Details
2. Course Description, Objectives, and Contents
3. Contents
4. Examples of Operating Systems

1. Administrative Details
2. Course Description, Objectives, and Contents
3. Contents
4. Examples of Operating Systems

Useful Info.

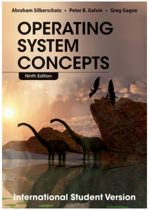
Instructor	Mohamad Jaber
Office	Bliss Hall, 312A
Phone	4256
Email	mj54@aub.edu.lb
Office Hours	MW 9:00-10:30AM, or by appointment

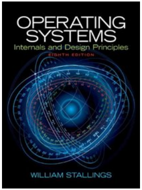
Teaching Assistant	Julia El Zini
Office	Bliss Graphics Lab
Email	jwe04@mail.aub.edu
Office Hours	MW 3:00-4:15PM or by appointment

Resources

The course will use the following two textbooks as references:

- Operating System Concepts, 9/e, Silberschatz, Galvin, and Gagne, Wiley (http://os-book.com) - assigned for the course
- Operating Systems: Internals & Design Principles, 8/e, William Stallings, Prentice Hall, (http://williamstallings.com/OperatingSystems/)

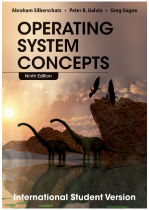

Abraham Silberschatz • Peter B. Galvin • Greg Gagne
OPERATING SYSTEM CONCEPTS
Ninth Edition
International Student Version



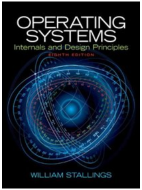
Resources

The course will use the following two textbooks as references:

- Operating System Concepts, 9/e, Silberschatz, Galvin, and Gagne, Wiley (http://os-book.com) - assigned for the course


Abraham Silberschatz • Peter B. Galvin • Greg Gagne
OPERATING SYSTEM CONCEPTS
Ninth Edition
International Student Version

- Operating Systems: Internals & Design Principles, 8/e, William Stallings, Prentice Hall, (http://williamstallings.com/OperatingSystems/)



Remark The course also uses some online materials and publications

Prerequisites

Prerequisites

The course assumes that you are comfortable with programming in C under Linux

Prerequisites

Prerequisites

The course assumes that you are comfortable with programming in C under Linux

Remark If not, please refer to Moodle or online materials - Do not hesitate to ask me!

Evaluation

Participation and Attendance	5%
Assignments and Projects	30%
Midterm (October 23 - Tentative)	30%
Final Exam (TBA)	35%
Total	100%

Evaluation

Participation and Attendance	5%
Assignments and Projects	30%
Midterm (October 23 - Tentative)	30%
Final Exam (TBA)	35%
Total	100%

- If you miss the midterm you should present a medical excuse and the final will count for 65%.

Evaluation

Participation and Attendance	5%
Assignments and Projects	30%
Midterm (October 23 - Tentative)	30%
Final Exam (TBA)	35%
Total	100%

- If you miss the midterm you should present a medical excuse and the final will count for 65%.
- The midterm and the final exam are cumulative, i.e., it will cover all class material to date.

Assignments/Projects

- Some of the assignments will not be graded; but I will ask questions about them during the exams or small quizzes.

Assignments/Projects

- Some of the assignments will not be graded; but I will ask questions about them during the exams or small quizzes.
- Although you are encouraged to discuss assignments, all work handed in must be done independently.
- Copying from each other will result in a zero grade for both copies.

Assignments/Projects

- Some of the assignments will not be graded; but I will ask questions about them during the exams or small quizzes.
- Although you are encouraged to discuss assignments, all work handed in must be done independently.
- Copying from each other will result in a zero grade for both copies.
- The minimum grade of assignments is **40**, except for plagiarism it is zero.

Assignments/Projects

- Some of the assignments will not be graded; but I will ask questions about them during the exams or small quizzes.
- Although you are encouraged to discuss assignments, all work handed in must be done independently.
- Copying from each other will result in a zero grade for both copies.
- The minimum grade of assignments is **40**, except for plagiarism it is zero.
- Your code must be easy to read:
  - Indent code, keep lines and (when possible) functions short.
  - Use a uniform coding style.
  - Put comments on structure members, globals, functions.

Attendance

- You are expected to attend lectures.

Be aware of AUB regulations regarding attendance!

Attendance

- You are expected to attend lectures.
- Lectures are in sequence.

Be aware of AUB regulations regarding attendance!

1. Administrative Details
2. Course Description, Objectives, and Contents
3. Contents
4. Examples of Operating Systems

Course Description and Objectives

- To provide an introduction to the fundamentals of operating system function, design, and implementation.

Course Description and Objectives

- To provide an introduction to the fundamentals of operating system function, design, and implementation.
- The course divides into three major sections:

Course Description and Objectives

- To provide an introduction to the fundamentals of operating system function, design, and implementation.
- The course divides into three major sections:
  - Concurrency (processes, threads, scheduling, synchronization, and deadlocks).

Course Description and Objectives

- To provide an introduction to the fundamentals of operating system function, design, and implementation.
- The course divides into three major sections:
  - Concurrency (processes, threads, scheduling, synchronization, and deadlocks).
  - Memory management (memory management strategies and virtual memory management).

Course Description and Objectives

- To provide an introduction to the fundamentals of operating system function, design, and implementation.
- The course divides into three major sections:
  - Concurrency (processes, threads, scheduling, synchronization, and deadlocks).
  - Memory management (memory management strategies and virtual memory management).
  - File systems, Secondary Storage, I/O systems.

Course Description and Objectives

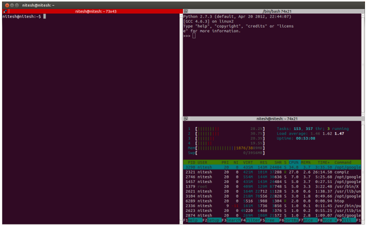
- To provide an introduction to the fundamentals of operating system function, design, and implementation.
- The course divides into three major sections:
  - Concurrency (processes, threads, scheduling, synchronization, and deadlocks).
  - Memory management (memory management strategies and virtual memory management).
  - File systems, Secondary Storage, I/O systems.

Understanding the OS makes you a more effective programmer!

Learning Outcomes

Upon completion of this course, you will be able to:

1. List the objectives and functions of modern Operating Systems.



```
nitesh@nitesh:~$
```

```
Python 2.7.3 (default, Apr 20 2012, 22:44:07)
[GCC 4.6.3] on linux2
Type "help", "copyright", "credits" or "licens
e" for more information.
>>>
```

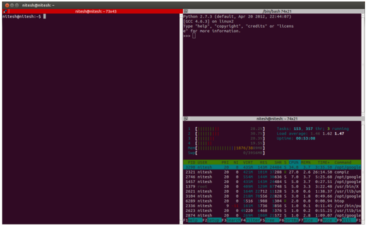
```
  1  [||||||||                 28.2%]     Tasks: 153, 357 thr; 3 running
  2  [|||||||||                30.7%]     Load average: 1.46 1.62 1.47
  3  [|||||||                  20.3%]     Uptime: 00:53:08
  4  [||||||                   19.5%]
  Mem[|||||||||||||||||1876/3889MB]
  Swp[                     0/3956MB]

  PID USER      PRI  NI  VIRT   RES   SHR S CPU% MEM%   TIME+  Command
 3290 nitesh     20   0  435M  143M 24484 S 34.0  3.7  3:35.50 /opt/google
 2321 nitesh     20   0  421M  101M 37288 R 27.0  2.6 26:14.50 compiz
 2746 nitesh     20   0  554M  144M 38636 S  7.0  3.7  5:25.68 /opt/google
 5457 nitesh     20   0  435M  143M 24484 S  5.0  3.7  0:27.51 /opt/google
 1379 root       20   0  409M  129M 87748 S  5.0  3.3  3:22.48 /usr/bin/X
 2621 nitesh     20   0  164M 22712 11128 S  3.0  0.6  1:38.37 /usr/lib/un
 3104 nitesh     20   0  218M 70556 22828 S  3.0  1.8  0:49.66 /opt/google
 6289 nitesh     20   0  5516  1988  1304 R  2.0  0.0  0:00.94 htop
 2336 nitesh      9 -11  161M  5736  3856 S  1.0  0.1  0:11.45 /usr/bin/pu
 2623 nitesh     20   0 49220  6468  3376 S  1.0  0.2  0:55.25 /usr/lib/in
 2874 nitesh     20   0  269M  108M 21572 S  1.0  2.8  1:09.07 /opt/google
F1Help  F2Setup F3SearchF4FilterF5Tree  F6SortByF7Nice -F8Nice +F9Kill  F1
```

Learning Outcomes

Upon completion of this course, you will be able to:

1. List the objectives and functions of modern Operating Systems.
2. Describe the concepts of processes, threads.


nitesh@nitesh: ~
nitesh@nitesh: ~ 73x43
/bin/bash 74x21
nitesh@nitesh: ~ 74x21
F1Help F2Setup F3Search F4Filter F5Tree F6SortBy F7Nice - F8Nice + F9Kill F1

```
nitesh@nitesh:~$
```

```
Python 2.7.3 (default, Apr 20 2012, 22:44:07)
[GCC 4.6.3] on linux2
Type "help", "copyright", "credits" or "licens
e" for more information.
>>>
```

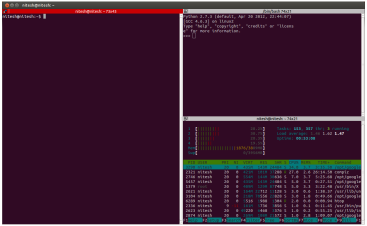
```
  1  [||||||||||               28.2%]    Tasks: 153, 357 thr; 3 running
  2  [||||||||||               30.7%]    Load average: 1.46 1.62 1.47
  3  [||||||||                 20.3%]    Uptime: 00:53:08
  4  [|||||||                  19.5%]
  Mem[||||||||||||||||||1876/3889MB]
  Swp[                      0/3956MB]

  PID USER      PRI  NI  VIRT   RES   SHR S CPU% MEM%   TIME+  Command
 3290 nitesh     20   0  435M  143M 24484 S 34.0  3.7  3:35.50 /opt/google
 2321 nitesh     20   0  421M  101M 37288 R 27.0  2.6 26:14.50 compiz
 2746 nitesh     20   0  554M  144M 38636 S  7.0  3.7  5:25.68 /opt/google
 5457 nitesh     20   0  435M  143M 24484 S  5.0  3.7  0:27.51 /opt/google
 1379 root       20   0  409M  129M 87748 S  5.0  3.3  3:22.48 /usr/bin/X
 2621 nitesh     20   0  164M 22712 11128 S  3.0  0.6  1:38.37 /usr/lib/un
 3104 nitesh     20   0  218M 70556 22828 S  3.0  1.8  0:49.66 /opt/google
 6289 nitesh     20   0  5516  1988  1304 R  2.0  0.0  0:00.94 htop
 2336 nitesh      9 -11  161M  5736  3856 S  1.0  0.1  0:11.45 /usr/bin/pu
 2623 nitesh     20   0 49220  6468  3376 S  1.0  0.2  0:55.25 /usr/lib/in
 2874 nitesh     20   0  269M  108M 21572 S  1.0  2.8  1:09.07 /opt/google
```

Learning Outcomes

Upon completion of this course, you will be able to:

1. List the objectives and functions of modern Operating Systems.
2. Describe the concepts of processes, threads.
3. Build and modify Operating System kernel from source code.



```
nitesh@nitesh:~$
```

```
Python 2.7.3 (default, Apr 20 2012, 22:44:07)
[GCC 4.6.3] on linux2
Type "help", "copyright", "credits" or "licens
e" for more information.
>>>
```

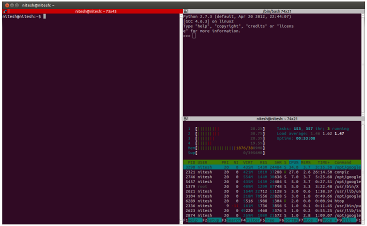
```
  1  [||||||||||             28.2%]     Tasks: 153, 357 thr; 3 running
  2  [||||||||||             30.7%]     Load average: 1.46 1.62 1.47
  3  [||||||||               20.3%]     Uptime: 00:53:08
  4  [|||||||                19.5%]
  Mem[||||||||||||||||1876/3889MB]
  Swp[                  0/3956MB]

  PID USER      PRI  NI  VIRT   RES   SHR S CPU% MEM%   TIME+  Command
 3290 nitesh     20   0  435M  143M 24484 S 34.0  3.7  3:35.50 /opt/google
 2321 nitesh     20   0  421M  101M 37288 R 27.0  2.6 26:14.50 compiz
 2746 nitesh     20   0  554M  144M 38636 S  7.0  3.7  5:25.68 /opt/google
 5457 nitesh     20   0  435M  143M 24484 S  5.0  3.7  0:27.51 /opt/google
 1379 root       20   0  409M  129M 87748 S  5.0  3.3  3:22.48 /usr/bin/X
 2621 nitesh     20   0  164M 22712 11128 S  3.0  0.6  1:38.37 /usr/lib/un
 3104 nitesh     20   0  218M 70556 22828 S  3.0  1.8  0:49.66 /opt/google
 6289 nitesh     20   0  5516  1988  1304 R  2.0  0.0  0:00.94 htop
 2336 nitesh      9 -11  161M  5736  3856 S  1.0  0.1  0:11.45 /usr/bin/pu
 2623 nitesh     20   0 49220  6468  3376 S  1.0  0.2  0:55.25 /usr/lib/in
 2874 nitesh     20   0  269M  108M 21572 S  1.0  2.8  1:09.07 /opt/google
F1Help  F2Setup F3SearchF4FilterF5Tree  F6SortByF7Nice -F8Nice +F9Kill  F1
```

Learning Outcomes

Upon completion of this course, you will be able to:

1. List the objectives and functions of modern Operating Systems.
2. Describe the concepts of processes, threads.
3. Build and modify Operating System kernel from source code.
4. Demonstrate practical experience under the Linux Operating System.



```
nitesh@nitesh:~$
```

```
Python 2.7.3 (default, Apr 20 2012, 22:44:07)
[GCC 4.6.3] on linux2
Type "help", "copyright", "credits" or "licens
e" for more information.
>>>
```

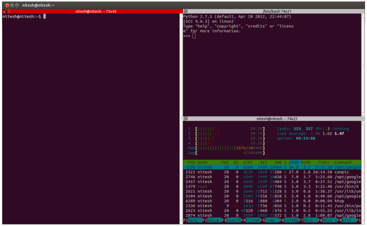
```
  1  [|||||||||                 28.2%]     Tasks: 153, 357 thr; 3 running
  2  [|||||||||                 30.7%]     Load average: 1.46 1.62 1.47
  3  [|||||||                   20.3%]     Uptime: 00:53:08
  4  [||||||                    19.5%]
  Mem[|||||||||||||||||1876/3889MB]
  Swp[                     0/3956MB]

  PID USER      PRI  NI  VIRT   RES   SHR S CPU% MEM%   TIME+  Command
 3290 nitesh     20   0  435M  143M 24484 S 34.0  3.7  3:35.50 /opt/google
 2321 nitesh     20   0  421M  101M 37288 R 27.0  2.6 26:14.50 compiz
 2746 nitesh     20   0  554M  144M 38636 S  7.0  3.7  5:25.68 /opt/google
 5457 nitesh     20   0  435M  143M 24484 S  5.0  3.7  0:27.51 /opt/google
 1379 root       20   0  409M  129M 87748 S  5.0  3.3  3:22.48 /usr/bin/X
 2621 nitesh     20   0  164M 22712 11128 S  3.0  0.6  1:38.37 /usr/lib/un
 3104 nitesh     20   0  218M 70556 22828 S  3.0  1.8  0:49.66 /opt/google
 6289 nitesh     20   0  5516  1988  1304 R  2.0  0.0  0:00.94 htop
 2336 nitesh      9 -11  161M  5736  3856 S  1.0  0.1  0:11.45 /usr/bin/pu
 2623 nitesh     20   0 49220  6468  3376 S  1.0  0.2  0:55.25 /usr/lib/in
 2874 nitesh     20   0  269M  108M 21572 S  1.0  2.8  1:09.07 /opt/google
F1Help  F2Setup F3SearchF4FilterF5Tree  F6SortByF7Nice -F8Nice +F9Kill  F1
```

Learning Outcomes

Upon completion of this course, you will be able to:

1. List the objectives and functions of modern Operating Systems.
2. Describe the concepts of processes, threads.
3. Build and modify Operating System kernel from source code.
4. Demonstrate practical experience under the Linux Operating System.
5. Construct multi-tasking programs and outline standard scheduling algorithms.



```
nitesh@nitesh:~$
```

```
Python 2.7.3 (default, Apr 20 2012, 22:44:07)
[GCC 4.6.3] on linux2
Type "help", "copyright", "credits" or "licens
e" for more information.
>>>
```

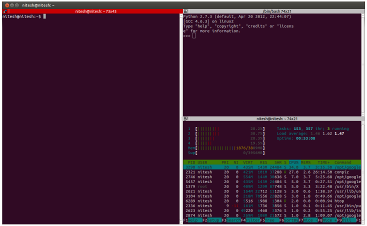
```
  1  [||||||||                28.2%]     Tasks: 153, 357 thr; 3 running
  2  [||||||||                30.7%]     Load average: 1.46 1.62 1.47
  3  [||||||                  20.3%]     Uptime: 00:53:08
  4  [|||||                   19.5%]
  Mem[|||||||||||||||    1876/3889MB]
  Swp[                        0/3956MB]

  PID USER      PRI  NI  VIRT   RES   SHR S CPU% MEM%   TIME+  Command
 3290 nitesh     20   0  435M  143M 24484 S 34.0  3.7  3:35.50 /opt/google
 2321 nitesh     20   0  421M  101M 37288 R 27.0  2.6 26:14.50 compiz
 2746 nitesh     20   0  554M  144M 38636 S  7.0  3.7  5:25.68 /opt/google
 5457 nitesh     20   0  435M  143M 24484 S  5.0  3.7  0:27.51 /opt/google
 1379 root       20   0  409M  129M 87748 S  5.0  3.3  3:22.48 /usr/bin/X
 2621 nitesh     20   0  164M 22712 11128 S  3.0  0.6  1:38.37 /usr/lib/un
 3104 nitesh     20   0  218M 70556 22828 S  3.0  1.8  0:49.66 /opt/google
 6289 nitesh     20   0  5516  1988  1304 R  2.0  0.0  0:00.94 htop
 2336 nitesh      9 -11  161M  5736  3856 S  1.0  0.1  0:11.45 /usr/bin/pu
 2623 nitesh     20   0 49220  6468  3376 S  1.0  0.2  0:55.25 /usr/lib/in
 2874 nitesh     20   0  269M  108M 21572 S  1.0  2.8  1:09.07 /opt/google
F1Help  F2Setup F3SearchF4FilterF5Tree  F6SortByF7Nice -F8Nice +F9Kill  F1
```

Learning Outcomes

Upon completion of this course, you will be able to:

1. List the objectives and functions of modern Operating Systems.
2. Describe the concepts of processes, threads.
3. Build and modify Operating System kernel from source code.
4. Demonstrate practical experience under the Linux Operating System.
5. Construct multi-tasking programs and outline standard scheduling algorithms.
6. Develop pseudo-code implementation to deal with deadlock.



```
nitesh@nitesh:~$
```

```
Python 2.7.3 (default, Apr 20 2012, 22:44:07)
[GCC 4.6.3] on linux2
Type "help", "copyright", "credits" or "licens
e" for more information.
>>>
```

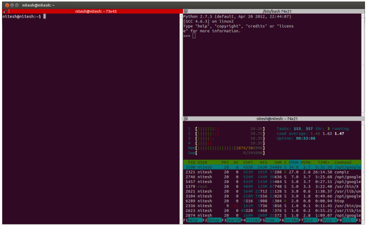
```
  1  [|||||||||              28.2%]     Tasks: 153, 357 thr; 3 running
  2  [|||||||||              30.7%]     Load average: 1.46 1.62 1.47
  3  [|||||||                20.3%]     Uptime: 00:53:08
  4  [||||||                 19.5%]
  Mem[||||||||||||||||1876/3889MB]
  Swp[                  0/3956MB]

  PID USER      PRI  NI  VIRT   RES   SHR S CPU% MEM%   TIME+  Command
 3290 nitesh     20   0  435M  143M 24484 S 34.0  3.7  3:35.50 /opt/google
 2321 nitesh     20   0  421M  101M 37288 R 27.0  2.6 26:14.50 compiz
 2746 nitesh     20   0  554M  144M 38636 S  7.0  3.7  5:25.68 /opt/google
 5457 nitesh     20   0  435M  143M 24484 S  5.0  3.7  0:27.51 /opt/google
 1379 root       20   0  409M  129M 87748 S  5.0  3.3  3:22.48 /usr/bin/X
 2621 nitesh     20   0  164M 22712 11128 S  3.0  0.6  1:38.37 /usr/lib/un
 3104 nitesh     20   0  218M 70556 22828 S  3.0  1.8  0:49.66 /opt/google
 6289 nitesh     20   0  5516  1988  1304 R  2.0  0.0  0:00.94 htop
 2336 nitesh      9 -11  161M  5736  3856 S  1.0  0.1  0:11.45 /usr/bin/pu
 2623 nitesh     20   0 49220  6468  3376 S  1.0  0.2  0:55.25 /usr/lib/in
 2874 nitesh     20   0  269M  108M 21572 S  1.0  2.8  1:09.07 /opt/google
F1Help  F2Setup F3SearchF4FilterF5Tree  F6SortByF7Nice -F8Nice +F9Kill  F1
```

Learning Outcomes

Upon completion of this course, you will be able to:

1. List the objectives and functions of modern Operating Systems.
2. Describe the concepts of processes, threads.
3. Build and modify Operating System kernel from source code.
4. Demonstrate practical experience under the Linux Operating System.
5. Construct multi-tasking programs and outline standard scheduling algorithms.
6. Develop pseudo-code implementation to deal with deadlock.
7. Describe and implement system memory management algorithms.



```
nitesh@nitesh:~$
```

```
Python 2.7.3 (default, Apr 20 2012, 22:44:07)
[GCC 4.6.3] on linux2
Type "help", "copyright", "credits" or "licens
e" for more information.
>>>
```

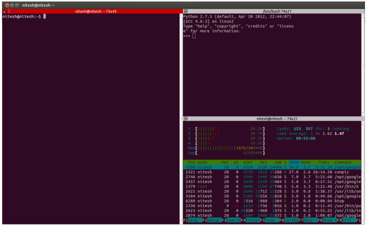
```
  1  [|||||||||                28.2%]     Tasks: 153, 357 thr; 3 running
  2  [|||||||||                30.7%]     Load average: 1.46 1.62 1.47
  3  [|||||||                  20.3%]     Uptime: 00:53:08
  4  [||||||                   19.5%]
  Mem[||||||||||||||||||1876/3889MB]
  Swp[                      0/3956MB]

  PID USER      PRI  NI  VIRT   RES   SHR S CPU% MEM%   TIME+  Command
 3290 nitesh     20   0  435M  143M 24484 S 34.0  3.7  3:35.50 /opt/google
 2321 nitesh     20   0  421M  101M 37288 R 27.0  2.6 26:14.50 compiz
 2746 nitesh     20   0  554M  144M 38636 S  7.0  3.7  5:25.68 /opt/google
 5457 nitesh     20   0  435M  143M 24484 S  5.0  3.7  0:27.51 /opt/google
 1379 root       20   0  409M  129M 87748 S  5.0  3.3  3:22.48 /usr/bin/X
 2621 nitesh     20   0  164M 22712 11128 S  3.0  0.6  1:38.37 /usr/lib/un
 3104 nitesh     20   0  218M 70556 22828 S  3.0  1.8  0:49.66 /opt/google
 6289 nitesh     20   0  5516  1988  1304 R  2.0  0.0  0:00.94 htop
 2336 nitesh      9 -11  161M  5736  3856 S  1.0  0.1  0:11.45 /usr/bin/pu
 2623 nitesh     20   0 49220  6468  3376 S  1.0  0.2  0:55.25 /usr/lib/in
 2874 nitesh     20   0  269M  108M 21572 S  1.0  2.8  1:09.07 /opt/google
F1Help  F2Setup F3SearchF4FilterF5Tree  F6SortByF7Nice -F8Nice +F9Kill  F1
```

Learning Outcomes

Upon completion of this course, you will be able to:

1. List the objectives and functions of modern Operating Systems.
2. Describe the concepts of processes, threads.
3. Build and modify Operating System kernel from source code.
4. Demonstrate practical experience under the Linux Operating System.
5. Construct multi-tasking programs and outline standard scheduling algorithms.
6. Develop pseudo-code implementation to deal with deadlock.
7. Describe and implement system memory management algorithms.
8. Describe file systems from disk management to high-level operations.



```
nitesh@nitesh:~$
```

```
Python 2.7.3 (default, Apr 20 2012, 22:44:07)
[GCC 4.6.3] on linux2
Type "help", "copyright", "credits" or "licens
e" for more information.
>>>
```

```
  1  [||||||||                28.2%]     Tasks: 153, 357 thr; 3 running
  2  [||||||||                30.7%]     Load average: 1.46 1.62 1.47
  3  [||||||                  20.3%]     Uptime: 00:53:08
  4  [|||||                   19.5%]
  Mem[||||||||||||||||1876/3889MB]
  Swp[                   0/3956MB]

  PID USER      PRI  NI  VIRT   RES   SHR S CPU% MEM%   TIME+  Command
 3290 nitesh     20   0  435M  143M 24484 S 34.0  3.7  3:35.50 /opt/google
 2321 nitesh     20   0  421M  101M 37288 R 27.0  2.6 26:14.50 compiz
 2746 nitesh     20   0  554M  144M 38636 S  7.0  3.7  5:25.68 /opt/google
 5457 nitesh     20   0  435M  143M 24484 S  5.0  3.7  0:27.51 /opt/google
 1379 root       20   0  409M  129M 87748 S  5.0  3.3  3:22.48 /usr/bin/X
 2621 nitesh     20   0  164M 22712 11128 S  3.0  0.6  1:38.37 /usr/lib/un
 3104 nitesh     20   0  218M 70556 22828 S  3.0  1.8  0:49.66 /opt/google
 6289 nitesh     20   0  5516  1988  1304 R  2.0  0.0  0:00.94 htop
 2336 nitesh      9 -11  161M  5736  3856 S  1.0  0.1  0:11.45 /usr/bin/pu
 2623 nitesh     20   0 49220  6468  3376 S  1.0  0.2  0:55.25 /usr/lib/in
 2874 nitesh     20   0  269M  108M 21572 S  1.0  2.8  1:09.07 /opt/google
F1Help  F2Setup F3SearchF4FilterF5Tree  F6SortByF7Nice -F8Nice +F9Kill  F1
```

1 Administrative Details

2 Course Description, Objectives, and Contents

3 Contents

4 Examples of Operating Systems

Contents

- Computer Architecture (quick review)
- Operating System Introduction
- Tutorial on Linux

Contents

- Computer Architecture (quick review)
- Operating System Introduction
- Tutorial on Linux
- Process Concept (Process, PCB, operations on processes, IPC, etc.)
- Multithreaded-Programming (*POSIX Threads Programming*)
- Process Scheduling
- Synchronization: critical section problem, semaphores, etc.
- Deadlocks

Contents

- Computer Architecture (quick review)
- Operating System Introduction
- Tutorial on Linux
- Process Concept (Process, PCB, operations on processes, IPC, etc.)
- Multithreaded-Programming (*POSIX Threads Programming*)
- Process Scheduling
- Synchronization: critical section problem, semaphores, etc.
- Deadlocks
- Memory Management
- Virtual Memory

Contents

- Computer Architecture (quick review)
- Operating System Introduction
- Tutorial on Linux
- Process Concept (Process, PCB, operations on processes, IPC, etc.)
- Multithreaded-Programming (*POSIX Threads Programming*)
- Process Scheduling
- Synchronization: critical section problem, semaphores, etc.
- Deadlocks
- Memory Management
- Virtual Memory
- File and I/O Systems

Computer Architecture

- Computer elements (Processor, Memory, I/O, etc.)
- Program execution
- Interrupts

Operating System Introduction

- Operating System Structure
- System calls
- Multi-programming, time-sharing, job scheduling
- Physical and virtual Memory
- Installation and booting of an Operating System

Process Concept

A program in execution, which forms the basis of all computation

- Process Scheduling
- Operations on Processes (creating, deleting, etc.)
- Interprocess Communication IPC

Multithreaded Programming

A fundamental unit of CPU utilization that forms the basis of multithreaded computer systems

- Multithreading model overview
- POSIX Threads Programming

Scheduling

- Introduce process scheduling, which is the basis for multiprogrammed operating systems
- Describe various process-scheduling algorithms

Synchronization

- The critical-section problem (*n* processes all competing to use some shared data)
- Peterson's solution
- Synchronization hardware
- Semaphores
- Classic problem of synchronization

Deadlocks

- Several processes may compete for a finite number of resources.
  1. *A* requires printer and is in possession of scanner!
  2. *B* requires scanner and is in possession of printer!

Deadlocks

- Several processes may compete for a finite number of resources.
  1. *A* requires printer and is in possession of scanner!
  2. *B* requires scanner and is in possession of printer!

- That is, process *A* waits for process *B* and process *B* waits for process *A*

Deadlocks

- Several processes may compete for a finite number of resources.
  1. *A* requires printer and is in possession of scanner!
  2. *B* requires scanner and is in possession of printer!
- That is, process *A* waits for process *B* and process *B* waits for process *A*
- Discuss different methods for handling deadlocks:
  - Deadlock prevention
  - Deadlock avoidance
  - Deadlock detection
  - Recovery from deadlock

Memory Management

- We must keep several processes in memory.
- This drastically improve the speed of the computer!
- Various memory-management techniques:
  - Swapping
  - Segmentation
  - Paging

Virtual Memory

Virtual memory is a technique that allows the execution of a processes that are not completely in memory

- Describe the benefits of a virtual memory system
- Explain the concepts of demand paging, page-replacement algorithms, etc.

File and I/O systems

- File system provides mechanism for storage and access to file contents (including data and programs).
    - Explain the function of file systems
    - File-system design and implementation

File and I/O systems

- File system provides mechanism for storage and access to file contents (including data and programs).
  - Explain the function of file systems
  - File-system design and implementation
- I/O vary widely in their function and speed (mouse, hard disk, etc.). Thus, varied method are needed to control them.
  - How to manage I/O operation and devises
  - Application I/O interface
  - Device drivers (if time permits)

1 Administrative Details

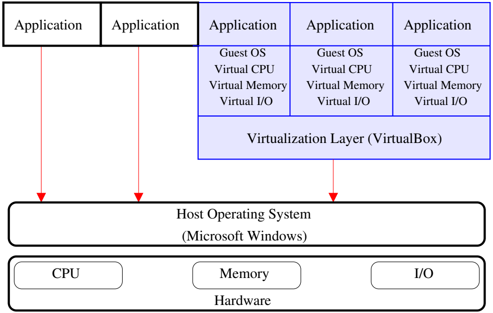
2 Course Description, Objectives, and Contents

3 Contents

4 Examples of Operating Systems

Popular Modern Operating Systems

- Examples: Linux (open source), Microsoft Windows, etc.
- During this course, we consider Ubuntu Linux. Ubuntu is popular Linux distribution.
- ***Instructions for installing Ubuntu inside Windows or Mac OS X using VirtualBox on Moodle.*** (Assignment zero)



History of Linux

- Unix (not free) was developed in 1969 by Bell Labs.

History of Linux

- Unix (not free) was developed in 1969 by Bell Labs.
- In 1984, Richard Stallman (researcher at MIT) created GNU project (a recursive acronym for “GNU’s Not Unix”!).
- GNU is free and open-source operating system that works like UNIX (same commands/API).

History of Linux

- Unix (not free) was developed in 1969 by Bell Labs.
- In 1984, Richard Stallman (researcher at MIT) created GNU project (a recursive acronym for “GNU’s Not Unix”!).
- GNU is free and open-source operating system that works like UNIX (same commands/API).
- Mac OS X is also based on UNIX. However, Windows is totally different.

History of Linux

- Unix (not free) was developed in 1969 by Bell Labs.
- In 1984, Richard Stallman (researcher at MIT) created GNU project (a recursive acronym for “GNU’s Not Unix”!).
- GNU is free and open-source operating system that works like UNIX (same commands/API).

- Mac OS X is also based on UNIX. However, Windows is totally different.

- In 1991, Linus Torvalds, a student at Helsinki University (Finland), was partly influenced by a speech about GNU in Finland in 1990.
- Linus began creating his own operating system (Linux = kernel) during his free time.

History of Linux

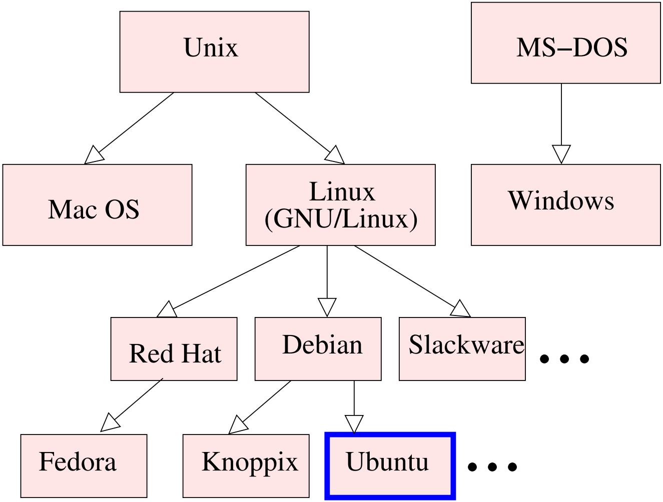
- Unix (not free) was developed in 1969 by Bell Labs.
- In 1984, Richard Stallman (researcher at MIT) created GNU project (a recursive acronym for “GNU’s Not Unix”!).
- GNU is free and open-source operating system that works like UNIX (same commands/API).

- Mac OS X is also based on UNIX. However, Windows is totally different.

- In 1991, Linus Torvalds, a student at Helsinki University (Finland), was partly influenced by a speech about GNU in Finland in 1990.
- Linus began creating his own operating system (Linux = kernel) during his free time.

- GNU includes: `gcc`, `gdb`, bash shell, GNU Core Utilities (`cat`, `ls`, `rm`, `cp`, etc.).
- GNU + Linux = GNU/Linux = **Linux**.

Operating Systems



Kernel

The kernel is the core of the operating system:

- Interrupt handlers
- Scheduler to share processor
- Memory management
- File systems
- Inter-process communication
- Networking
- . . .

Example of Linux kernel version (major version . minor version . revision).

Latest Stable Release:

4.12.9