

CSE2031 Software Tools - Introduction

Summer 2010

Przemyslaw Pawluk

Department of Computer Science and Engineering
York University
Toronto

May 4, 2010

Notes

Your Instructor

- Przemyslaw (Pshemo) Pawluk
- Lectures: Tuesday 6.00-8.00pm in CSE1006
- Lab: Tuesday 4.00-6.00pm
- Office hours:
 - CSEB 2053 (Database lab.)
 - Tuesday, Thursday 5-6pm
- email: pawluk@cse.yorku.ca

Notes

CSE 2031 - Content

This course introduces software tools that are used for building applications and in the software development process. Furthermore, you will be exposed to the layers between a programming language and the operating system and the CPU. The course covers the following topics:

- ANSI-C (C Basics, stdio, pointers, memory management, overview of ANSI-C libraries)
- Shell programming under Unix (Bourne shell, filters and pipes)
- Testing and debugging

All the above topics will be applied in practical programming assignments.

Notes

CSE 2031 - Objectives

By the end of the course, you should be able to

- Write modest-sized programs in C
- Test and debug C code
- Write programs using UNIX shell scripting language
- Use Unix utilities for fast solving practical problems.

Notes

Readings

Text book

- *The C Programming Language*, Brian Kernighan and Dennis Ritchie 2nd edition, Prentice-Hall
- *Practical Programming in the Unix Environment*, Edited by W. Strzlinger, Pearson

Other useful reading material

UNIX Shells by Example Author. Ellie Quigley 4th ED Publisher: Prentice-Hall

Notes

CSE2031 - Format

Web

<http://www.cse.yorku.ca/course/2031>

Lectures

Try to do assigned reading (please see calendar on course web site)

Lab

Prism Lab. - where we will be available for you to help with Ex's and other issues

Notes

Grading

Grading is test and assignment based

- Written Tests 50%
 - Midterm 20%
 - Final 30%
- Lab tests 20%
 - Midterm 10%
 - Final 10%
- Assignments: 30%
 - A1 10%
 - A2 10%
 - A3 10%

Notes

Missed Work

Deadlines

Late work will not be accepted!

If you miss deadline:

- You have to provide doctor's note
- The weight of assignment or Mid-term will be added to the Final

Notes

Submissions

Work from home

You can use Putty or any other client to work from home.
Putty is available on the course webpage.

Connection

- Server: red.cse.yorku.ca
- Connection type: SSH
- Use your prism login and password

In case of any problems contact our technicians (see:
<http://www.cse.yorku.ca/cspeople/staff/index.html> for details)

Important!

Your submissions have to execute correctly in Prism Lab!

Notes

Coding style

CSE2031
Software
Tools - In-
troduction
Przemyslaw
Pawluk

About this
course

ANSI-C
Unix and
Shell

Software
Develop-
ment
Process

Introduction
to C

Summary

11 / 45

Comments, names, indentation

Use proper indentation and names in your code to make it readable. Part of your mark will depend upon your coding style.

Information about standards and conventions

http://www.cse.yorku.ca/course_archive/2009-10/S/2031/conventions.html

Notes

Why C?

CSE2031
Software
Tools - In-
troduction
Przemyslaw
Pawluk

About this
course

ANSI-C

Unix and
Shell

Software
Develop-
ment
Process

Introduction
to C

Summary

13 / 45

Powerful

A widely used general purpose programming language with high-level constructs and ability to handle low-level activities (direct memory access, memory allocation, BitWise operations, etc).

Efficient and fast

It produces efficient programs.

Predecessor of modern oo languages

Many languages derived from C (e.g., C++, Java)

Good to learn

A good language to learn testing and debugging (C has poor error detection and significantly fewer safeguards than Java)

Notes

Why Unix?

CSE2031
Software
Tools - In-
troduction
Przemyslaw
Pawluk

About this
course

ANSI-C

Unix and
Shell

Software
Develop-
ment
Process

Introduction
to C

Summary

15 / 45

Widely used

Widely used operating system with time-sharing, multi-tasking, and multi-user. First written in assembler in 1969, rewritten in C in 1973 – portable!

Performance

- stability,
- security and
- predictability

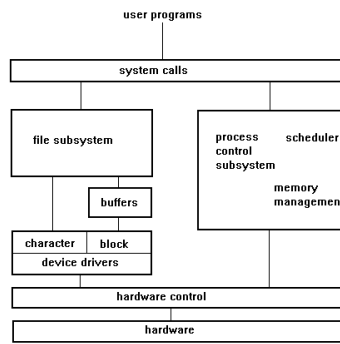
Idea used in many other systems

Many systems derived from it

- Linux is a clone of Unix
- embedded OS

Notes

Good to know—Unix Structure



Notes

Unix Philosophy

Make a program do one thing, but do it well
cat, more, mv, mkdir, ls, pwd, wc, grep, cut

Use existing utilities to solve larger problems
grep 999999 marks.txt—mail -s CSE2031-A1 pawluk@gmail.com

Expect output of every program to be usable by another program
Simple text interface.

Notes

Software Development Cycle

Specification

- What not How!
- Is the law.
- No change unless it is approved.

Design

- Is based on specification
- Algorithm (i.e., method, how to do the job).
- Data structures

Implementation

Sometimes called coding or programming. It is based on design.

Notes

Software Development Cycle 2

Testing

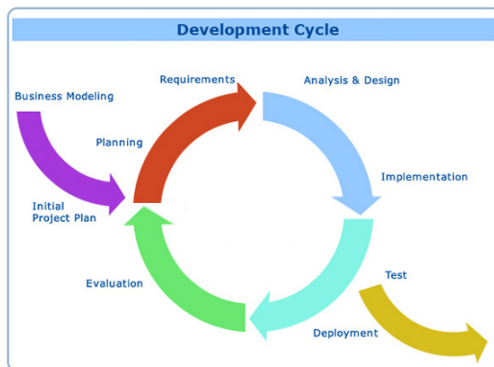
- Checks if your program conforms to the specification.
- Test cases
- Done by programmers, testers and customers

Debugging

Run when testing fails to find where the problem is and fix it.

Notes

Software Development Cycle 3



Notes

Why Testing?

- 1990 AT&T long distance calls fail for 9 hours
 - Wrong location for C break statement
- 1996 Ariane rocket explodes on launch
 - Overflow converting 64-bit float to 16-bit integer
- 1999 Mars Climate Orbiter crashes on Mars
 - Missing conversion of English units to metric units
- Therac: A radiation therapy machine that delivered massive amount of radiations killing at least 5 people
 - Among many others, the reuse of software written for a machine with hardware interlock. Therac did not have hardware interlock.

Notes

Dijkstra

Testing can show the presence of faults, not their absence

Notes

- Java-like (Actually Java has a C-like syntax), some differences
- No //, only /* */ multi line and no nesting
- No garbage collection
- No classes
- No exceptions (try catch)
- No type strings

Notes

```
#include <stdio.h>
```

```
int course;
```

```
int main()
```

```
{  
    course=2031;  
    printf( "CSE" );  
    printf( "%d\n", course );  
    printf( "press any key" );  
    getchar();  
    return 0;  
}
```

Includes

include information about libraries used by your programme

Globals

Global variables – try to avoid this kind of declarations

Functions and Procedures

Function *main* receives no arguments and returns int. You can define subprogrammes here.

Notes

Program compilation and execution

Compilation

To compile program run the command

```
cc example.c
```

Execution

To execute your program run the executable produced by compiler called a.out by typing

```
./a.out
```

Notes

Variables' Types

Simple

- int
- float
- char
- short
- long
- double

Compounds

- array
- structure
- union

Other

- pointer – is an address of some space in the memory
- function – can be also passed as an argument to some other function or be returned as a result of the function

Notes

Mixed type arithmetic

Operands	Result	Example
int int	int	$3/2 = 1$
float float	float	$3.0/2.0 = 1.5$
int float	float	$3/2.0 = 1.5$

Notes

Cast

```
int varA = 9, varB = 2;
double varC;

varC = varA / varB; /* varC is 4.0 */
varC = varA / (double) varB; /* varC is 4.5 */
```

Notes

Precedence

()	Parentheses	L to R	1
++, --	Postincrement	L to R	2
++, --	Preincrement	R to L	3
+, -	Positive, negative	L to R	3
*, /, %	Multiplication, division	L to R	4
+, -	Addition, subtraction	L to R	5
<=, >=, >, <	Relational operator	L to R	6
==, !=	Relational operator	L to R	7
&&	Logical AND	L to R	8
	Logical OR	L to R	9
+, =, ++, *, =, /, %, =	Compound assignment	R to L	10
=	Assignment	R to L	10

Notes

Constructs in C

Control flow

- decision making (branching)
 - if-else
 - switch
- looping
 - while , for
 - do
 - early exit from the loop break

Notes

Boolean expressions

Relational operators

`==, !=, <, <=, >, >=`

Logical operators

`&&, ||, !`

True-False

False is 0

Anything else is True

Notes

Conditional expressions

Test ? if-true:if-false

If the Test is true then if-true is returned otherwise if-false

Example

`z=(a>b)? a : b`

Notes

I/O

I/O of the program

- Every program has a standard input and output (stdin, stdout and stderr)
- Usually, keyboard and monitor

Char by char

`int getchar();` `int putchar(int c);`

Formatted I/O

<code>printf("Test no. %d\n", x);</code>	<code>%d</code>	integer
	<code>%s</code>	string
<code>scanf("%x%d", &x, &y);</code>	<code>%c</code>	character
	<code>%f</code>	float
	<code>%lf</code>	double precision

Notes

Preprocessor directives

include

```
#include <file.h>
```

Replaces this declaration with the content of *file.h*

define

```
#define abc xyz
```

Replaces each occurrence of *abc* with *xyz*

Notes

Pre- vs. Post-

Preincrement

```
++x
```

Postincrement

```
x++
```

What will be printed?

```
...
a=b=10;
y=a++;
z=++b;
printf("y=%d, z=%d", y, z);
...
```

Notes

Expressions

How can you express *d* and *x*?

```
...
a+=10; //a=a+10
b*=5; //b=b*5
c/=++b; //c=c/(b+1)
d-=b++; //???
x*=y+1; //???
...
```

Notes

- Decimal: 456
- Octal (starts with 0): 0710
- Hexadecimal (starts with 0x or 0X): 0x1C8
- Different notations:
 - 7L for long int =7
 - 8U for unsigned
 - For floats 24, 23.45, 123.45e-8, 3.4F, 2.15L

Notes

- You must open the file before you read or write to it (what about stdin,).
- The system checks the file, and returns a small non-negative integer known as file descriptor, all reads and writes are through this file descriptor.
- 0,1,2 are reserved for stdin, stdout, and stderr.

Notes

Example

```
FILE *fp;  
int char;  
//FILE *fopen(char *name, char *mode)  
fp=fopen(" test.txt", " r+a" );  
...  
char = fgetc(fp);  
fputc(char, fp);  
...  
if (fp)  
fclose(fp);
```

Notes

Files-operations

CSE2031
Software
Tools - In-
troduction
Przemyslaw
Pawluk

About this
course

ANSI-C

Unix and
Shell

Software
Develop-
ment
Process

Introduction
to C

Summary

42 / 45

Operations

- fopen - Name is a character string containing the name of the file, mode is a character string to indicate how the file will be used, and Mode could be r, w, a, r+b,
- fgetc - reads a character from the file
- fputc - writes a character to the file (where?)
- fclose - closes file pointed by fp

Notes

Summary

CSE2031
Software
Tools - In-
troduction
Przemyslaw
Pawluk

About this
course

ANSI-C

Unix and
Shell

Software
Develop-
ment
Process

Introduction
to C

Summary

44 / 45

- 1 About this course
- 2 ANSI-C
- 3 Unix and Shell
- 4 Software Development Process
- 5 Introduction to C
- 6 Summary

Notes

For the next lecture

CSE2031
Software
Tools - In-
troduction
Przemyslaw
Pawluk

About this
course

ANSI-C

Unix and
Shell

Software
Develop-
ment
Process

Introduction
to C

Summary

45 / 45

- Testing

Notes
