CS32 Summer 2013

Intro, Unix-like OS, Scripting

Victor Amelkin

August 8, 2013

Plan for Today

- Administrativia
- Unix-like OS
- Basics of GNU/Linux
- Scripting
- Programming Assignment 1

Plan for Today

- Administrativia
- Unix-like OS
- Basics of GNU/Linux
- Scripting
- Programming Assignment 1

General Information

- **TA:** Victor Amelkin
- **Email:** victor+cs32@cs.ucsb.edu
- **Web:** http://cs.ucsb.edu/~victor/ta/cs32/
- **Office Hours:** Mondays, 1-3pm, GSL
- **Forum:** https://piazza.com/ucsb/summer2013/cs32
- **Main Web-page:** https://www.cs.ucsb.edu/~koc/cs32/

Plan for Quarter

- **Quarter:** August 5 – September 12
- **Discussions:** Thursdays, 3:30-4:50pm
- **Programming Assignments:** released weekly (5 PAs)
  - PA1 has been released (will talk about it later)
  - Work in pairs; need to form teams today
- **Homeworks:** TBA (3 HWs)
- **Project:** released during Week 3, due during Week 6
  - Work in pairs
- **Midterm**

Grading

- This course is about programming:
  - Programming Assignments: **35%**
  - Project: **30%**
  - Midterm: 20%
  - Homeworks: 15%
- No curving
- Late submissions: (not recommended)
  - PAs/HWs: -20% per day after the deadline
  - Project: no late submissions

Some Rules

- Always sign your code
- You cannot submit not your own code
  - If you want to use some off-the-shelf implementation (not standard C/C++), ask in advance
- You cannot share your code with other students
- You cannot work on other students' assignments
- You can discuss general ideas with other students
- If you do not know what a particular bit of code does, you cannot turn it in. **Be always ready to “defend” your code**
- If in doubt, ask

Plan for Today

- Administrativia
- Unix-like OS
- Basics of GNU/Linux
- Scripting
- Programming Assignment 1

What you will need (part 1)

- College of Engineering account to access csil.cs.ucsb.edu
  - No account → create ASAP: https://accounts.engr.ucsb.edu/create/

- Unix-like OS
  - GNU/Linux (Ubuntu, Fedora, Mint, Arch, …)
  - Mac OS X
  - BSD

- Most UCSB's machines run on Fedora (formerly, Red Hat)

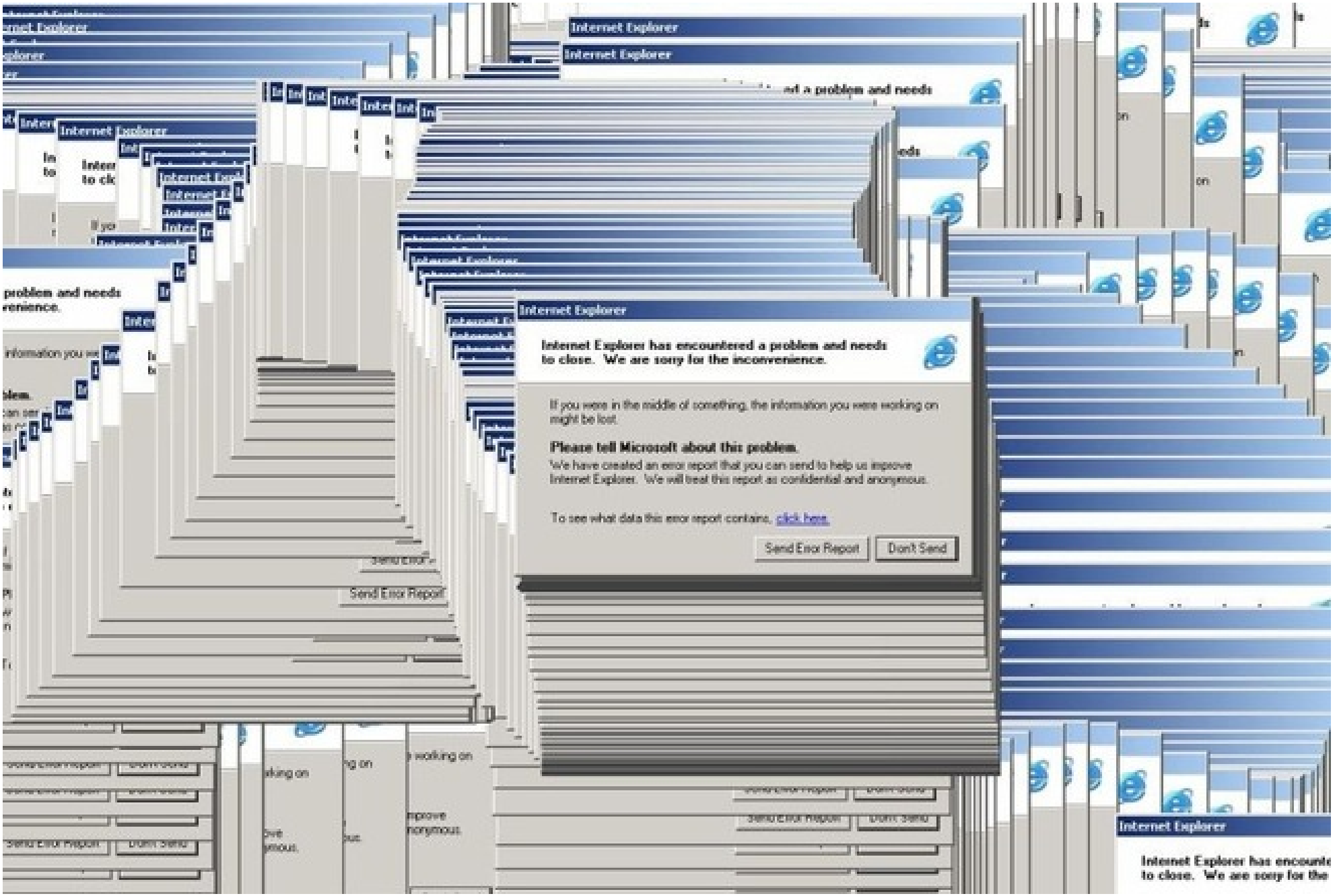
What you will need (part 2)

- Tools:
  - Unix tools (bash, ssh, grep, …)
  - text editor (vim or emacs) or an IDE (eclipse)
  - compiler (g++)
  - debugger (gdb)

- Optional:
  - profiler (gprof)
  - source control (git or hg or svn) – for the project

- See http://cs.ucsb.edu/~victor/ta/cs32/ for useful links

“What if I use Windows...”



Solutions for the Windows problem

- Work at CSIL with Fedora
- Use `putty` (and `Xming`) to connect to csil.cs.ucsb.edu from your machine and then use GNU/Linux
  - "Remotely working with CSIL via SSH from Windows" http://cs.ucsb.edu/~victor/ta/cs32/ssh_csil/
- Install Cygwin or MinGW+MSYS on your Windows
- Install GNU/Linux on your machine (at least in Virtual Box)

Plan for Today

- Administrativia
- Unix-like OS
- Basics of GNU/Linux
- Scripting
- Programming Assignment 1

Minimum Command/Tool Set

- Files/dirs: `ls, pwd, cd, mkdir, rm, rmdir, cp, mv, ln, find, which`
- Viewing files: `cat, less, head, tail`
- Text editing: `vim, emacs`
- File text manipulation: `grep, cut, sort, sed`
- File properties: `file, chmod, chown`
- Processes: `fg, bg, jobs; ps, top, kill`
- Network: `ssh, wget, scp`
- Dev-tools: `nm, ldd, strings; gcc, g++, gdb, gprof`
- Misc: `tar, diff, finger, screen`

Good video tutorials: [link]

Best command ever: **man**

Network

- **ssh** – connect to a host via SSH
  - `ssh victor@csil.cs.ucsb.edu` (basic)
  - `ssh -X victor@csil.cs.ucsb.edu` (with X11 forwarding)
  - `ssh csil` (with SSH config)

  `~/.ssh/config`

```
Host csil
    HostName csil.cs.ucsb.edu
    User victor
    ForwardX11 yes
```

- **wget** – download a document via HTTP to the current dir
  - `wget http://cs.ucsb.edu/~victor/ta/cs32/pa/1/pa1.tar.gz`
- **scp** – copy a file via SSH
  - `ssh hw.tar.gz victor@csil.cs.ucsb.edu:~/cs32/hw1/`

Working with Files/Dirs

- **ls** – list files in the current dir
  - ls (basic)
  - ls -acg (list all entries with extra info)
- **pwd** – print the path to the current dir
- **cd** – change current dir
- **mkdir**, **rmdir** – create/delete dir
- **rm** – remove file(s)
  - rm *.jpg (delete all jpgs in current dir)
  - rm -rf ./somedir/ (delete dir somedir and its contents)
- **cp**, **mv** – copy/move
- **ln** – create symbolic link
- **find**, **which** – search for files

Tar + GZip

- **Pack** and compress file1, file2, file3 into myarchive.tar.gz

```
tar czf myarchive.tar.gz file1 file2 file3
```

- **Unpack** myarchive.tar.gz to ./dir/

```
tar xf myarchive.tar.gz ./dir/
```

Processes and Jobs

- **jobs** – list current jobs
- **fg %i** – move I'th job to foreground
- **bg %i** – move I'th job to background
- **ps** – list current processes
- **top** – same, but interactive
- **kill** – kill a process

```
xclock (run xclock or any other program)
Ctrl+Z (switch to shell)
jobs (list active jobs)
bg %1 (move job xclock to background)
kill %1 (kill xclock by its job index)
or
ps -A | grep 'xclock' (learn xclock's PID)
kill 15651 (kill xclock by its PID)
```

Viewing Files + File Properties

- **cat** – print file contents to standard output
- **less** – similar to cat, but prints less
- **head** – print a few initial lines of a file
- **tail** – print a few last lines of a file
- **file** – prints file type

- **chown** – change file's owner
  - `chown newowner ./file1`
  - `chown -hR newowner ./dir` (recursively)

- **chmod** – change file permissions
  - `chmod u+rwx ./file` – add **r**ead-**w**rite-e**x**ecute permissions for current **u**ser
  - `chmod g-wx ./file` – revoke **g**roup's **w**rite-e**x**ecute permissions

  Tutorial on Unix permissions: [link]

Pipes

- Feeding output of one command as input to another command:

```
echo 'Hello wc command!' | wc -w

man finger | grep 'BSD' | tail -n 1
```

- Tutorial on pipes: [link]

Redirecting Output to File

```
finger coke@cs.cmu.edu > coke.info
cat coke.info
```

```
One entry found for exact uid match
Login: coke                    Name: Drink Coke
Directory: /afs/cs.cmu.edu
No Plan
```

Redirecting File to Standard Input

```
echo “hello, world” > ./info

wc -w < ./info
   2 (number of words in file ./info)

cat ./info | wc -w
   2 (number of words in file ./info)
```

Plan for Today

- Administrativia
- Unix-like OS
- Basics of GNU/Linux
- Scripting
- Programming Assignment 1

Scripting Basics

- `bash` is our script interpreter
- Script files: `myscript.sh`
- Script starts with shebang `#!{path}`
- Scripts must be executable

```
chmod u+x ./myscript.sh
```

- Example: shebang

  http://cs.ucsb.edu/~victor/ta/cs32/disc1/shebang/

Executing Linux Commands

- You can execute Linux commands from your script:

```
#!/bin/bash
pwd
mkdir newdir
cd newdir
# > - rewrites; >> - appends
echo 'hellooooo' >> newfile
echo ' world!!!' >> newfile
cd ..
```

- Example: basic

  http://cs.ucsb.edu/~victor/ta/cs32/disc1/basic/

Variables

```
#!/bin/bash

myvar1=100
myvar2=200
myvar3="luvbash"

echo $myvar1
echo $(($myvar1 + $myvar2 + 17))
echo "First variable is $myvar3!"
echo $myvar1 + $myvar3
echo 'Here, $myvar is not substituted (thanks to
single quotes).'
```

- Example: argvar

  http://cs.ucsb.edu/~victor/ta/cs32/disc1/argvar/

Command-Line Arguments

```
#!/bin/bash

echo "input args: $* (-- all of them)"

echo "first actual arg: $0 (-- always present;
path to the script)"

echo "first input arg: $1"
echo "second input arg: $2"

echo "number of input args: $#"
```

- Example: argvar

  http://cs.ucsb.edu/~victor/ta/cs32/disc1/argvar/

More on Pipes

- trim.sh

```
#!/bin/bash
echo $(echo $1 | sed -e 's/^ *//g' -e 's/ *$//g')
```

- exec.sh

```
#!/bin/bash
username="victor"
fullname_raw=$( finger victor | head -n 1 | cut -d':' -f3 )
echo "fullname_raw = '$fullname_raw'."

fullname=$( ./trim.sh "$fullname_raw" )
echo "Full name of '$username' is '$fullname'."
```

- Example: execio (exec, trim)

  http://cs.ucsb.edu/~victor/ta/cs32/disc1/execio/

Complex I/O Redirect

```
#!/bin/bash

username=$1
if [ $# -eq 0 ]
then
   echo "Supply the username."
   exit 1
fi

result=$(finger -ms $username 2>&1 1>/dev/null | wc -l)
if [ $result -eq 0 ]
then
   echo "User exists."
else
   echo "User does not exist."
fi
```

- Example: execio (userexists)

  http://cs.ucsb.edu/~victor/ta/cs32/disc1/execio/

Conditionals

```
#!/bin/bash

if [ -z $1 ]; then
   echo “First input argument is empty.”
fi


if [ -f $2 ]; then
   echo “Second input argument is a path to an
existing file.”
fi


cd
if [ -r "./public_html/index.html" ]; then
   echo "My home page exists and is readable"
fi
```

Loops

```
#!/bin/bash

for dir in $(find ./mydir -maxdepth 1 -type d)
do
   echo $dir
done
```

- See also: [link]

Plan for Today

- Administrativia
- Unix-like OS
- Basics of GNU/Linux
- Scripting
- Programming Assignment 1

Programming Assignment 1

- Released:

  http://cs.ucsb.edu/~victor/ta/cs32/

- Due: August 15, 11:59pm

- Overview