15213 Recitation Section C

Shimin Chen
Sept. 9, 2002

Outline

- Introduction
- Unix and C
- Playing with Bits
- Practice Problems

Introducing Myself

- Try to pronounce my name:

  Shimin Chen
- My office hour:

  Wed 2-3pm, WeH 8019
- Contact:
  - Email: chensm@cs.cmu.edu
  - Phone: x8-5143

Unix and C

- Getting to know Makefile
- PATH environment
- Common pitfalls of C programming

Makefile: writing rules

- Rule Format:

  targets : *prerequisites*

  tab → *command ...*
- Example:

```
btest: btest.c bits.c decl.c tests.c btest.h bits.h
	gcc -O -Wall -o btest bits.c btest.c decl.c tests.c
```

Makefile: using variables

```
CC = gcc
CFLAGS = -O -Wall

btest: btest.c bits.c decl.c tests.c btest.h bits.h
	$(CC) $(CFLAGS) -o btest bits.c btest.c decl.c tests.c
```

- The value of a variable is the string after the "="

L1 Makefile

```
# Student's Makefile for the CS:APP Data Lab
TEAM = ac00
VERSION = 1
HANDINDIR = /afs/cs.cmu.edu/academic/class/15213-f02/L1/handin

CC = gcc
CFLAGS = -O -Wall

btest: btest.c bits.c decl.c tests.c btest.h bits.h
	$(CC) $(CFLAGS) -o btest bits.c btest.c decl.c tests.c

handin:
	cp bits.c $(HANDINDIR)/$(TEAM)-$(VERSION)-bits.c

clean:
	rm -f *.o btest
```


Makefile Reference

- GNU Make Manual
  - Do a google search for "GNU Make Manual"
  - http://www.gnu.org/manual/make/html_node/make_toc.html

PATH environment

- *What is PATH?*
  - The directories to search for executable commands
- *How come the Unix shell can't find my program "btest"?*
  - "." is not in the PATH
  - ./btest
  - Add "." into PATH:
  - http://www-2.cs.cmu.edu/afs/cs/academic/class/15213-f02/www/

Pitfalls of C Programming (1)

```
main ()
{
   ….
   for (int i=0; i<10; i++) {
      ……
   }
   ….
}
```

Local Variables should be declared at the beginning of the procedure

Declaring Local Variables

```
main ()
{int i;
   ….
   for (i=0; i<10; i++) {
      ……
   }
   ….
}
```


Pitfalls of C Programming (2)

- No cout, cin!
- Use printf, scanf

```
#include <stdio.h>
main ()
{ int i = 15213;
  float f = 1.5213;
  char str[20] = "15213";

   printf ("integer: %d\n", i);
   printf ("float: %f\n", f);
   printf ("string: %s\n", str);
}
```


Pitfalls of C Programming (3)

```
#include <stdio.h>
main ()
{int i;
 char str[20];

      scanf ("%d", i);
      scanf ("%s", str);
}
```

scanf requires an address

Using Addresses to Call scanf

```
#include <stdio.h>
main ()
{int i;
 char str[20];

      scanf ("%d", &i);
      scanf ("%s", str);
}
```


Pitfalls of C Programming (4)

- No "new", "delete" operators!
- Use "malloc" and "free"

```
#include <malloc.h>
aProcedure ()
{char *buffer;
        buffer = (char *)malloc (4096);
        if (buffer ==NULL) {
                printf ("can't allocate memory!\n");
                exit (1);
        }
        ……..
        free (buffer);
}
```


Turn on the Warnings

- Compile with "-Wall"
- Check and fix the warnings

Playing with Bits

- Powers of 2
- Binary, Hexadecimal, Decimal
- Unsigned and Two's Complement

Powers of 2

- Let's write down 2^0 ~ 2^{16}

Hexadecimal Digits

- 0~9, A, B, C, D, E, F
- What are the corresponding decimals?
- What are the corresponding binary numbers?

Hexadecimal ç Ł Binary

- Let's convert $(3B6D)_{16}$ into Binary:
- Answer: $(11101101101101)_2$
- Let's convert $(1100010010010011)_2$ into hexadecimal
- Answer: $(C493)_{16}$

Binaryç Ł Decimal

- Let's convert $(101010)_2$ to decimal
- Answer: 42
- Let's convert $(37)_{10}$ to binary
- Answer: $(100101)_2$

Unsigned and Two's Complement

- 4-bit integer
- What are the decimals for the following unsigned representations?

  0000, 1111, 0101, 1000, 0111, 1011

- What are the decimals if they are two's complements?

Practice Problem (1)

- Negate the following 8-bit two's complement:

  X_1=01000100 $-X_1$=?
  X_2=10011000 $-X_2$=?
  X_3=00000000 $-X_3$=?
  X_4=10000000 $-X_4$=?
- Complement then add 1

Practice Problem (2)

- Extract a bit from an integer:

```
int extract_a_bit (int x, int pos)
{
        ???
}
```

e.g. extract_a_bit (2, 1) = 1
extract_a_bit (2, 5) = 0

A Solution

```
int extract_a_bit (int x, int pos)
{
        return ((x>>pos)&1);
}
```

Practice Problem (3)

- Compute the bit parity of an integer. Return 1 if there are odd number of 1s; return 0 if there are even number of 1s.

```
int bit_parity (int x)
{
        ???
}
```


A Solution

```
int bit_parity (int x)
{
        int word16 = x ^ (x>>16);
        int word8 = word16 ^ (word16 >> 8);
        int word4 = word8 ^ (word8 >> 4);
        int word2 = word4 ^ (word4 >> 2);
        int bit = word2 ^ (word2 >> 1);
        return bit & 1;
}
```


Common Questions of L1

- X << 32 doesn't work!
  - Some compilers produce: X << (shift&0x1f)
- Right shift:
  - Logical for unsigned int
  - Arithmetic (sign extended) for signed int
    - (not standard, but almost all compilers do this)