Introduction to Computer Science and Programming in C

Session 6: September 18, 2008

Columbia University

Announcements

- Reminder: Homework 1 is out. Due Tuesday
- TA: Sharath Avadoot Gururaj.
  Office hours Monday 1:30-3:30. UNI: sa2617
- WiCS: Contact Rebecca Collins
  Rebecca.L.C@gmail.com
- If you need to email questions, send to all **four** of us for fastest response:
  `{bert@cs, ds2664@, yl2505@, sa2617@}columbia.edu`

Review

- Initialization, truncation, increment/decrement
- **Arrays**: blocks of **indexed** variables
- **Strings:** Arrays of characters
  - string.h – strlen(), strcopy()
- Input during program: fgets(), sscanf()
- Command line input: int argc, char *argv[]

Today

- Algorithmic tools
  - Conditionals: **if**, **else**
  - Loops: **while** loops, **for** loops

Motivation

```
/*
multiply.c – Takes two integers as command line
arguments and displays their product
*/

#include <stdio.h>
int main(int argc, char *argv[])
{
    int a, b, c;

    sscanf(argv[1], “%d”, &a);
    sscanf(argv[2], “%d”, &b);
    c = a*b;
    printf(“%d times %d is %d\n”, a, b, c);
    return 0;
}
```

Motivation

- Only run this program if argc is equal to 3. Otherwise, tell the user that there was a mistake.
- “argc is equal to 3” is either true or false.

Motivation

```
#include <stdio.h>
int main(int argc, char *argv[])
{
    int a, b, c;

    if (argc==3) {    /* NOTE: double-equals == */
        sscanf(argv[1], "%d", &a);
        sscanf(argv[2], "%d", &b);
        c = a*b;
        printf("%d times %d is %d\n", a, b, c);
    } else {
        printf("Input error\n");
    }
    return 0;
}
```

if/else Syntax

- ```
  if (<condition>)
      <then do stuff>;
  else
      <then do something else>;
  ```

- When you have multiple lines, surround with brackets

  ```
  if (<condition>) {
      <then do stuff>;
      <do more stuff>;
  }
  ```
  ```

Conditional Operators

- Comparison operators:
  - Equality: a==b, a!=b
  - Inequalities: a<b, a>b, a<=b, a>=b
- AND: (a==b) && (b==c)
- OR: (a==b) || (b==c)
- NOT: !(a==b)

Boolean Logic

- Systematic reasoning about truth
- Named after George Boole
- We can treat each conditional clause as a **Boolean** variable.
  - 2 settings: TRUE (1) or FALSE (0)
- (C doesn't have a Boolean type. Use int)

Boolean Logic

- Consider Boolean variables: A, B, C
  - **A && B = B && A** (communative)
  - **A || (B || C) = (A || B) || C** (associative)
  - **A && (B || C) = (A && B) || (A && C)** (distributive)
  - **!(A && B) = !A || !B** (DeMorgan's Law)
- Obeys order of operations: && before ||
- Often analogous: && is like *, || is like +

Loops

- If we want to repeat a piece of code, use **loops**.

```
int population[50]; /* initializing without loops */
population[0] = 0;
population[1] = 0;
population[2] = 0;
...
population[48] = 0;
population[49] = 0;
```

- Start index at 0. While index is less than 50, set population at index to 0.

Loops: while

- We can use a **while** loop:

```
int population[50]; /* initializing without loops */
int index=0;
while (index<50) {
    population[index] = 0;
    index++;                /* increment shortcut */
}
```

- Syntax:

```
while (<condition>) {  /* Again, the brackets     */
    <statement(s)>;    /* are optional if you     */
}                      /* have only one statement */
```

Loops: for

- When loop condition depends on an index, **for** loops can be more useful.

- Syntax:

```
for (<initialization>; <condition>; <counting>) {
    <do stuff>
}
```

- ```
  int population[50], index;
  for (index = 0; index<50; index++) {
      population[index] = 0;
  }
  ```
  ```

Practice

- Initialize tictactoe

```
#include <stdio.h>

int main(int argc, char *argv[])
{
    int tictactoe[3][3];

    /* what's next? */

...
```

Loops

- Sometimes the choice of **for** versus **while** is stylistic.
- However, sometimes only **while** makes sense:

```
int sum = 0, input = 1;
char line[30];
while (input != 0) {
    printf("Enter a number (0 to quit): ");
    fgets(line, sizeof(line), stdin);
    sscanf(line, "%d\n", &input);
    sum = sum+input;
}
```

switch

```
if (month==1) {
    printf("Jan.");
} else if (month==2) {
    printf("Feb.");
} else if (month==3) {
    printf("Mar.");
} else if (month==4) {
    printf("Apr.");
} else if (month==5) {
    printf("May");
} else {
    printf("Summer");
}
```

```
switch(month) {
case 1:
    printf("Jan.");
    break;
case 2:
    printf("Feb.");
    break;
...
case 5:
    printf("May");
    break;
default:
    printf("Summer");
    break;
}
```

Reading

- Practical C Programming, Chapter (5), 6 and 8