CMSC 246 Systems Programming

Fall 2019
Bryn Mawr College

Sep 11

Loops

- The `while` statement has the form

  `while (` *expression* `)` *statement*

- General form of the `do` statement:

  `do` *statement* `while (` *expression* `) ;`

- General form of the `for` statement:

  `for (` *expr1* `;` *expr2* `;` *expr3* `)` *statement*

  expr1, *expr2*, and *expr3* are expressions.

- Example:

```
for (int i = 0; i < n; i++) {}
```

- Infinite loops

```
for(;;;) {}
```

```
while(1) {}
```

- Comma operator

```
for(int i=0, j=100; i<100; i++,j--)
```

Program: Convert Fahrenheit to Celsius

- The `c2ftable.c` program prompts the user to enter two Fahrenheit temperatures (lower and upper); it then prints the equivalent Celsius temperature, for each temperature starting from lower to upper, in increments of 10.
- Sample program output:

```
Enter Fahrenheit temperature range: 0 150
Fahrenheit Celsius
0                   -17.78
10                  -12.22
…                   …
150                 65.56
```

- The program will allow temperatures that aren’t integers.

How **`scanf`** Works

- Sample input:

  `1-20.3-4.0e3¤`

- The call of `scanf` is the same as before:

  `scanf("%d%d%f%f", &i, &j, &x, &y);`

- Here's how `scanf` would process the new input:
  - `%d`. Stores 1 into `i` and puts the – character back.
  - `%d`. Stores –20 into `j` and puts the `.` character back.
  - `%f`. Stores 0.3 into `x` and puts the – character back.
  - `%f`. Stores –4.0 × 103 into `y` and puts the new-line character back.

scanf peeking problems

```
#include <stdio.h>
#define FREEZING 32.0f
#define SCALE 1.8f
#define CC(v) (FREEZING + SCALE*v)

int main(void)
{
    float f, c;
    for (int i=0; i<100; i++)
    {
        printf("Enter a Celcius temperature: ");
       scanf("%f", &c);  // use %lf for double
       f = CC(c);
       printf("%d: Celcius: %.1f  Fahrenheit: %.1f\n", i, c, f);
    }
    return 0;
}
```

demo, then c2fsafe.c

Ordinary Characters in Format Strings

- When it encounters one or more white-space characters in a format string, `scanf` reads white-space characters from the input until it reaches a non-white-space character (which is "put back").
- When it encounters a non-white-space character in a format string, `scanf` compares it with the next input character.
  - If they match, `scanf` discards the input character and continues processing the format string.
  - If they don't match, `scanf` puts the offending character back into the input, then aborts.
    - Looping

Ordinary Characters in Format Strings

- Examples:
  - If the format string is "`%d/%d`" and the input is •`5/`
    - `96`, `scanf` succeeds.
    - If the input is •`5•/•96`, `scanf` fails, because the `/` in the format string doesn't match the space in the input.
- To allow (but not require) spaces around the /, use the format string "`%d / %d`" instead.

Program: Adding Fractions

- The `addfrac.c` program prompts the user to enter two fractions and then displays their sum.
- Sample program output:

```
Enter first fraction: 5/6
Enter second fraction: 3/4
The sum is 38/24
```

addfrac.c

```
/* Adds two fractions */

#include <stdio.h>

int main(void)
{
  int num1, denom1, num2, denom2, result_num, result_denom;

  printf("Enter first fraction: ");
  scanf("%d/%d", &num1, &denom1);

  printf("Enter second fraction: ");
  scanf("%d / %d", &num2, &denom2);

  result_num = num1 * denom2 + num2 *denom1;
  result_denom = denom1 * denom2;
  printf("The sum is %d/%d\n",result_num, result_denom)

  return 0;
}
```

Acknowledgements

Some content from these slides is based on the book, C Programming – A Modern Approach, By K. N. King, 2nd Edition, W. W. Norton 2008.

Some content is also included from the lecture slides provided by Prof. K. N. King. Thank You!