EECS 22: Advanced C Programming

Lecture 6

Rainer Dömer

doemer@uci.edu

The Henry Samueli School of Engineering
Electrical Engineering and Computer Science
University of California, Irvine

Lecture 6: Overview

- Review of the C Programming Language
  - Types
    - Basic types and typical ranges
    - Type conversion
    - Types in expressions
    - Type qualifiers
  - Application Example **`PhotoLab.c`**

Basic Types, Ranges

- Integer Types
  - **`char`** Character, e.g. **`'a'`**, **`'b'`**, **`'1'`**, **`'*'`**
    - typical range 8 bit **`[-128,127]`**
  - **`short int`** Short integer, e.g. **`-7`**, **`0`**, **`42`**
    - typical range 16 bit **`[-32768,32767]`**
  - **`int`** Integer, e.g. **`-7`**, **`0`**, **`42`**
    - typical range 32 bit **`[-2147483648,2147483647]`**
  - **`long int`** Long integer, e.g. **`-99L`**, **`9L`**, **`123L`**
    - typical range 64 bit (same as **`long long int`**)
  - **`long long int`** Very long integer, e.g. **`12345LL`**
    - typical range 64 bit **`[-9223372036854775808,9223372036854775807]`**
- Integer Types can be
  - **`signed`** negative and positive values (incl. 0)
  - **`unsigned`** positive values only (incl. 0)

Basic Types, Ranges

- Floating Point Types
  - **`float`** Floating point with single precision
    - Typical representation: 24 + 8 = 32 bit
    - Example: **`3.5f`**, **`-0.234f`**, **`10e8f`**
  - **`double`** Floating point with double precision
    - Typical representation: 53 + 11 = 64 bit
    - Example: **`3.5`**, **`-0.23456789012`**, **`10e88`**
  - **`long double`** Floating point with high precision
    - Typical representation: 113 + 15 = 128 bit
    - Example: **`12345678.123456e123L`**
- Remember:
  Most floating point values are *approximations only*!
  - Finite storage size can only store finite amount of numbers

Type Conversion

- Explicit Type Conversion
  - types can be explicitly converted to other types, by use of the type cast operator:
    `(type) expression`
  - the target type is named explicitly in parentheses before the source expression
  - Examples:
    - **`Float = (float) LongInt`**
      - converts the **`long int`** value into a **`float`** value
    - **`Integer = (int) Double`**
      - converts the **`double`** value into an **`int`** value
      - any fractional part is truncated!
    - **`Char = (char) LongLongInt`**
      - converts the **`long long int`** value into a **`char`** value
      - any out-of-range values are silently cut off!

Type Conversion

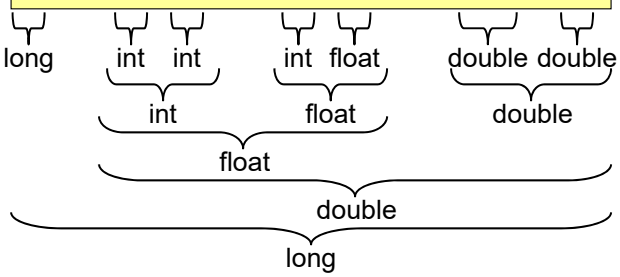
- Implicit Type Conversion
  - Type promotion
    - integral promotion
      - **`unsigned`** or **`signed char`** is promoted to **`unsigned`** or **`signed int`** before any operation
      - **`unsigned`** or **`signed short`** is promoted to **`unsigned`** or **`signed int`** before any operation
    - binary arithmetic operators are defined only for same types
      - the smaller type is converted to the larger type (before operation)
      - Examples:
        - » **`ShortInt * LongInt`** results in a **`long int`** type
        - » **`LongDouble * Float`** results in a **`long double`** type
  - Type coercion
    - most types are automatically converted to expected types
    - Example: **`Double = Float`**, or **`Char = LongInt`**

Types in Expressions

- Expressions are composed of constants, variables and operators, each of which has an associated type
- Example:

```
short int s;
int       i;
long int  l;
float     f;
double    d;

l  =   2 * s   +   i * f   -   0.5  * d;
```



Type Qualifiers

- Types may be further qualified
  - Type qualifier **`const`**
    - The value of a **`const`** object cannot be changed
    - Initialization is mandatory, assignment is not possible
    - Example:
      - **`const double pi = 3.1415926536;`**
    - ➢ Object may be placed in read-only memory (ROM)
  - Type qualifier **`volatile`**
    - The value of a **`volatile`** object must not be used for compiler optimizations
    - Machine registers for memory-mapped I/O are volatile
    - Example:
      - **`volatile char *StatusReg = 0x40000000;`**
      - **`while(*StatusReg == 0x00) ;`**
    - ➢ Accesses to **`volatile`** objects must not be optimized away

Application Example

- Program example: **`PhotoLab`**
  - Digital image manipulation
    - Read an image from a file
    - Manipulate the image in memory
    - Write the modified image to file
  - Portable Pixel Map (PPM) file format
    - simple uncompressed file format for color images
    - Header section (including picture width, height)
    - Data section (pixel values in Red/Green/Blue format)

```
P6
600 400
255
RGBRGBRGB…
```


Application Example

- Program example: **`PhotoLab.c`** (part 1/5)

```
/*********************************************************/
/* PhotoLab.c: assignment 2 for EECS 22 in Fall 2017     */
/*                                                       */
/* modifications: (most recent first)                    */
/* 09/27/17 RD    adjusted for lecture usage             */
/*********************************************************/

#include <stdio.h>
#include <stdlib.h>

/*** global definitions ***/

const int WIDTH  = 600;      /* image width */
const int HEIGHT = 400;      /* image height */
const int SLEN   =  80;      /* max. string length */

...
```

Application Example

- Program example: **`PhotoLab.c`** (part 2/5)

```
...
/*** function definitions ***/

/* write the RGB image to a PPM file    */
/* (return 0 for success, >0 for error) */

int SaveImage(const char Filename[SLEN],
              unsigned char R[WIDTH][HEIGHT],
              unsigned char G[WIDTH][HEIGHT],
              unsigned char B[WIDTH][HEIGHT])
{

  ...

} /* end of SaveImage */

...
```


Application Example

- Program example: **`PhotoLab.c`** (part 3/5)

```
...

/* read an image file into the RGB data structure */
/* (return 0 for success, >0 for error) */

int LoadImage(const char fname[SLEN],
              unsigned char R[WIDTH][HEIGHT],
              unsigned char G[WIDTH][HEIGHT],
              unsigned char B[WIDTH][HEIGHT])
{

  ...

} /* end of LoadImage */

...
```

Application Example

- Program example: **`PhotoLab.c`** (part 4/5)

```
...
/* modify the image...  ;-) */

void ModifyImage(unsigned char R[WIDTH][HEIGHT],
                 unsigned char G[WIDTH][HEIGHT],
                 unsigned char B[WIDTH][HEIGHT])
{   int  x, y;

    for(y=0; y<HEIGHT; y++)
    {   for(x=0; x<WIDTH; x++)
        {
            B[x][y] = (R[x][y] + G[x][y] + B[x][y]) / 5;
            R[x][y] = (unsigned char) (B[x][y]*1.6);
            G[x][y] = (unsigned char) (B[x][y]*1.6);
        }
    }

} /* end of ModifyImage */
...
```


Application Example

- Program example: **`PhotoLab.c`** (part 5/5)

```
...
/*** main program ***/

int main(void)
{
    unsigned char R[WIDTH][HEIGHT];
    unsigned char G[WIDTH][HEIGHT];
    unsigned char B[WIDTH][HEIGHT];

    if (LoadImage("HSSOE.ppm", R,G,B) != 0)
        { return 10; }
    ModifyImage(R, G, B);
    if (SaveImage("HSSOE1965.ppm", R,G,B) != 0)
        { return 10; }
    return 0;
} /* end of main */

/* EOF */
```

Application Example

- Example session: **`PhotoLab.c`**

```
% vi PhotoLab.c
% gcc PhotoLab.c -o PhotoLab -Wall –ansi –std=c99
% pnmtojpeg HSSOE.ppm > HSSOE.jpg
% PhotoLab
% pnmtojpeg HSSOE1965.ppm > HSSOE1965.jpg
%
```

HSSOE.ppm

HSSOE1965.ppm