EECS 22: Advanced C Programming

Lecture 9 (TuTh)

Rainer Dömer

doemer@uci.edu

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

Part 1: Overview

- Warm-up Quiz
- Course Administration
  - Midterm course evaluation
- Assertions
  - Using and disabling assertions

Quiz: Question 6

- Which of the following constructs is a valid binary operator in C?
  (Check all that apply!)
  a) `/`
  b) `%`
  c) `!`
  d) `@`
  e) `>>`

Quiz: Question 6

- Which of the following constructs is a valid binary operator in C?
  (Check all that apply!)
  a) `/`
  b) `%`
  c) `!`
  d) `@`
  e) `>>`

Quiz: Question 7

- What is the value of the integer **x** after the following statement?

```
x = 11 / 3 + 11 % 3;
```

a) **1**
b) **2**
c) **3**
d) **4**
e) **5**

Quiz: Question 7

- What is the value of the integer **x** after the following statement?

```
x = 11 / 3 + 11 % 3;
```

a) **1**
b) **2**
c) **3**
d) **4**
e) **5**

Quiz: Question 8

- What is the value of the variable **x** after the following lines of code?

```
unsigned char x = 42;

x += 1024;
if (x < 0)
   { x = 10; }
if (x > 255)
   { x = 20; }
```

a) **0**
b) **10**
c) **20**
d) **42**
e) **1066**

Quiz: Question 8

- What is the value of the variable **x** after the following lines of code?

```
unsigned char x = 42;

x += 1024;
if (x < 0)
   { x = 10; }
if (x > 255)
   { x = 20; }
```

a) **0**
b) **10**
c) **20**
d) **42**
e) **1066**

Quiz: Question 9

- Which of the following program fragments will *not* terminate? (Check all that apply!)

a)
```
int a = 1;
while(a < 1000000)
   { a++; }
```

b)
```
int a = 0;
while(a < 1000)
   { a = a * 3; }
```

c)
```
int a = 1;
while(a == 1)
   { a = a % 10; }
```

d)
```
int a = 1;
while(a < 1000)
   { a = a << 1; }
```

e)
```
int a = 10;
while(a > 0)
   { a = a / 3; }
```


Quiz: Question 9

- Which of the following program fragments will *not* terminate? (Check all that apply!)

a)
```
int a = 1;
while(a < 1000000)
   { a++; }
```

b)
```
int a = 0;
while(a < 1000)
   { a = a * 3; }
```

c)
```
int a = 1;
while(a == 1)
   { a = a % 10; }
```

d)
```
int a = 1;
while(a < 1000)
   { a = a << 1; }
```

e)
```
int a = 10;
while(a > 0)
   { a = a / 3; }
```

Quiz: Question 10

- Given two global variables **`int x=7`** and **`int y=8`**, which of the following functions properly swaps the values such that **x=8** and **y=7**?
  (Check all that apply!)

a)
```
void swap(int x, int y)
{ x = y; y = x;
}
```

b)
```
void swap(void)
{ x = y; y = x;
}
```

c)
```
void swap(void)
{ int t;
  t = x; x = y; y = t;
}
```

d)
```
void swap(void)
{ int t;
  t = y; y = x; x = t;
}
```

e)
```
void swap(int x, int y)
{ int t;
  t = x; x = y; y = t;
}
```


Quiz: Question 10

- Given two global variables **`int x=7`** and **`int y=8`**, which of the following functions properly swaps the values such that **x=8** and **y=7**?
  (Check all that apply!)

a)
```
void swap(int x, int y)
{ x = y; y = x;
}
```

b)
```
void swap(void)
{ x = y; y = x;
}
```

c)
```
void swap(void)
{ int t;
  t = x; x = y; y = t;
}
```

d)
```
void swap(void)
{ int t;
  t = y; y = x; x = t;
}
```

e)
```
void swap(int x, int y)
{ int t;
  t = x; x = y; y = t;
}
```

Course Administration

- Midterm Course Evaluation
  - One week, starting today!
  - Wednesday, Oct. 25, 8am – Nov. 1, 8am
  - Online via EEE Evaluation application
- Feedback from students to instructors
  - Completely voluntary
  - Completely anonymous
  - Very valuable
    - Help to improve this class!
- Mandatory Final Course Evaluation
  - expected for week 10 (TBA)

Assertions

- Run-time Checks for Diagnostics and Debugging
  - Can be manually implemented

```
...
#ifdef DEBUG
if (value > 100)
   { printf("Something's wrong, value is >100!");
     abort();
   } /* fi */
#endif /* DEBUG */
...
```

  - Can be enabled at time of compilation (for development)

```
% gcc prog.c -ansi -std=c99 -Wall -o prog -DDEBUG
%
```

  - Can be disabled at time of compilation (for final release)

```
% gcc prog.c -ansi -std=c99 -Wall -o prog
%
```

Assertions

- *Assertions*: Diagnostics by the standard C library

```
#include <assert.h>
...
assert(value <= 100);
```

  - Header file **`assert.h`**
    - Defines **`assert(condition)`** as a preprocessor macro
  - Assertion failure
    - At run-time, if **`condition`** evaluates to **`false`**, the program is aborted with a corresponding diagnostic message

```
assertion: prog.c:12: main: Assertion `value <= 100' failed.
Abort
```

  - Disabling assertions
    - If **`NDEBUG`** is defined when **`assert.h`** is included, the **`assert()`** macro has no effect (empty statement)

```
% gcc -DNDEBUG prog.c -o prog
%
```


Assertions

- Example: Square Root Calculation **`Root.c`**

```
#include <assert.h>

double Root(double x) /* square root approximation */
{   double l, m, r, d;

    assert(x >= 0.0); /* caller must supply positive x */
    l = 0.0; r = x;
    do{ m = l + (r-l)/2.0;
        d = m * m - x;
        if (d < 0.0)
          { d = -d;
            l = m; }
        else
          { r = m; }
      } while (d > 1e-10);
    return m;
}
```

  - ➢ Assertion protects the *contract* between caller and callee
    - Caller is in charge of ensuring positive argument to function call
    - Callee relies on this agreement (otherwise the loop will not terminate!)

Assertions

- Advise on Using Assertions
  - ➢ Use assertions often
    - Confirm assumptions about parameters, calculated values, etc.
    - Assertions are cheap (low run-time overhead)!
  - ➢ Use assertions in software development from the beginning
    - Diagnostic messages are very helpful in development
      - – Program aborts as soon as a value is out of expected range
      - – Location and problem condition are shown
    - This can avoid more serious problems later
  - ➢ Disable assertions for final program delivered to the user
    - Diagnostic messages are of no use to the end user!
      - – User has no idea about condition and source location
  - ➢ Beware of side-effects in assertions
    - Implemented as a macro!
    - Can lead to *Heisenbugs* which disappear when debugging is on!

Part 2: Overview

- Course Administration
  - – Fairness quiz
  - – Midterm course evaluation
- Midterm Course Review
  - – Syntax and semantics of C programs
  - – Types, expressions, statements, functions
  - – Recursion, modules, Makefile, debugging
- Practice
  - – Review Quiz

EECS 22 Fairness Quiz

- Given the clearly announced hard deadline, which of the following are valid excuses for acceptance of a late submission? (Check all that apply!)
  a) My watch showed I still had 2 minutes.
  b) I used the wrong submission command.
  c) I was still debugging the last problem in my code.
  d) My network connection broke down.
  e) I had a medical emergency and can provide documentation.

EECS 22 Fairness Quiz

- Given the clearly announced hard deadline, which of the following are valid excuses for acceptance of a late submission? (Check all that apply!)
  a) My watch showed I still had 2 minutes.
  b) I used the wrong submission command.
  c) I was still debugging the last problem in my code.
  d) My network connection broke down.
  e) I had a medical emergency and can provide documentation.

Course Administration

- Midterm Course Evaluation
  - This week!
  - Wednesday, Oct. 25, 8am – Nov. 1, 8am
  - Online via EEE Evaluation application
- Feedback from students to instructors
  - Completely voluntary
  - Completely anonymous
  - Very valuable
    - Help to improve this class!
- Mandatory Final Course Evaluation
  - expected for week 10 (TBA)

Midterm Course Review

- Introduction, course setup, Linux
- Tokens, basic types, operators, formatted I/O
- Control-flow statements, conditionals and loops
- Arrays and array indexing
- Functions, call graph, call trace, call stack
- Pass by value vs. pass by reference
- Recursion
- Scope, variable lifetime, storage classes
- Compiler components, translation units
- Make and Makefile, rules, targets and dependencies
- Assertions, debugging, GDB commands

Quiz: Question 11

- Today's computers run at which clock speed?
  a) 85 MPH
  b) 1 kHz
  c) 1 ms
  d) 1 GHz
  e) 1 MHz

Quiz: Question 11

- Today's computers run at which clock speed?
  a) 85 MPH
  b) 1 kHz
  c) 1 ms
  d) 1 GHz
  e) 1 MHz

Quiz: Question 12

- Which of the following names are valid keywords in ANSI C?
  (Check all that apply!)
  a) `if`
  b) `when`
  c) `void`
  d) `main`
  e) `Int`

Quiz: Question 12

- Which of the following names are valid keywords in ANSI C?
  (Check all that apply!)
  a) `if`
  b) `when`
  c) `void`
  d) `main`
  e) `Int`

Quiz: Question 13

- Which of the following names are valid identifiers in ANSI C?
  (Check all that apply!)
  a) `xyz`
  b) `PC`
  c) `dollar amount`
  d) `My_Very_Long_Variable_Name`
  e) `2fast4you`

Quiz: Question 13

- Which of the following names are valid identifiers in ANSI C?
  (Check all that apply!)
  a) `xyz`
  b) `PC`
  c) `dollar amount`
  d) `My_Very_Long_Variable_Name`
  e) `2fast4you`

Quiz: Question 14

- Which of the following constructs are valid type names in ANSI C?
  (Check all that apply!)
  a) `short char`
  b) `long double`
  c) `signed long long`
  d) `unsigned float`
  e) `signed`

Quiz: Question 14

- Which of the following constructs are valid type names in ANSI C?
  (Check all that apply!)
  a) `short char`
  b) `long double` ←
  c) `signed long long` ←
  d) `unsigned float`
  e) `signed` ←

Quiz: Question 15

- Which of the following constants is of type **`double`**?
  (Check all that apply!)
  a) **`42`**
  b) **`.42`**
  c) **`4e2`**
  d) **`4E2`**
  e) **`42f`**

Quiz: Question 15

- Which of the following constants is of type **`double`**?
  (Check all that apply!)
  a) **`42`**
  b) **`.42`**
  c) **`4e2`**
  d) **`4E2`**
  e) **`42f`**

Quiz: Question 16

- What is the value of the integer **x** after the following statement?

```
x = 3 << 2 >> 1;
```

a) **Syntax Error!**
b) **3**
c) **6**
d) **12**
e) **321**

Quiz: Question 16

- What is the value of the integer **x** after the following statement?

```
x = 3 << 2 >> 1;
```

a) **Syntax Error!**
b) **3**
c) **6**
d) **12**
e) **321**

Quiz: Question 17

- Which of the following expressions correctly computes the polynomial $p = 2x^2 - 3x + 4$? (Check all that apply!)

a) `p = 2x^2 - 3x + 4;`
b) `p = 2xx - 3x + 4;`
c) `p = x*x*2 - 3*x + 4.0;`
d) `p = 2*(x*x + 3)*x + 4;`
e) `p = (2*x - 3)*x + 4;`

Quiz: Question 17

- Which of the following expressions correctly computes the polynomial $p = 2x^2 - 3x + 4$? (Check all that apply!)

a) `p = 2x^2 - 3x + 4;`
b) `p = 2xx - 3x + 4;`
c) `p = x*x*2 - 3*x + 4.0;`
d) `p = 2*(x*x + 3)*x + 4;`
e) `p = (2*x - 3)*x + 4;`

Quiz: Question 18

- What is the result of the evaluation of the following expression?

```
1 == 2 || 3 < 4 && 5 > 6
```

a) **123456**
b) **true**
c) **false**
d) **1**
e) **0**

Quiz: Question 18

- What is the result of the evaluation of the following expression?

```
1 == 2 || 3 < 4 && 5 > 6
```

a) **123456**
b) **true**
c) **false**
d) **1**
e) **0**

Quiz: Question 19

- Simple prime number test:
  The following code fragment iterates variable *i* over the range $2 \le i < x$ to find a divisor of *x.*
  What should go into box 1 in line 4?
  a) `i = 0;`
  b) `i = 1;`
  c) `i = 2;`
  d) `i = x;`
  e) `x = 0;`

```
int x, i;
printf("Please input a number: ");
scanf("%d", &x);
initialize variable i
while(i < x)
{ if(x % i == 0)
  { printf("%d is not prime\n", x);
    break;
  }
  i++;
}
if( none of the i is a divisor of x )
  { printf("%d is prime\n", x);
  }
```


Quiz: Question 19

- Simple prime number test:
  The following code fragment iterates variable *i* over the range $2 \le i < x$ to find a divisor of *x.*
  What should go into box 1 in line 4?
  a) `i = 0;`
  b) `i = 1;`
  c) `i = 2;`
  d) `i = x;`
  e) `x = 0;`

```
int x, i;
printf("Please input a number: ");
scanf("%d", &x);
initialize variable i
while(i < x)
{ if(x % i == 0)
  { printf("%d is not prime\n", x);
    break;
  }
  i++;
}
if( none of the i is a divisor of x )
  { printf("%d is prime\n", x);
  }
```

Quiz: Question 20

- Simple prime number test:
  The following code fragment iterates variable *i* over the range $2 \le i < x$ to find a divisor of *x*.
  What should go into box 2 in line 12?
  a) `x / i == 0`
  b) `x < i`
  c) `i / x == 0`
  d) `i + 1 == x`
  e) `i == x`

```
int x, i;
printf("Please input a number: ");
scanf("%d", &x);
initialize variable i
while(i < x)
{ if(x % i == 0)
  { printf("%d is not prime\n", x);
    break;
  }
  i++;
}
if( none of the i is a divisor of x )
  { printf("%d is prime\n", x);
  }
```


Quiz: Question 20

- Simple prime number test:
  The following code fragment iterates variable *i* over the range $2 \le i < x$ to find a divisor of *x*.
  What should go into box 2 in line 12?
  a) `x / i == 0`
  b) `x < i`
  c) `i / x == 0`
  d) `i + 1 == x`
  e) `i == x`

```
int x, i;
printf("Please input a number: ");
scanf("%d", &x);
initialize variable i
while(i < x)
{ if(x % i == 0)
  { printf("%d is not prime\n", x);
    break;
  }
  i++;
}
if( none of the i is a divisor of x )
  { printf("%d is prime\n", x);
  }
```

Quiz: Question 21

- Which of the following variable declarations are valid in ANSI-C?
  (Check all that apply!)

a) `double xyz;`
b) `double x, y, z;`
c) `double x = 1.0;`
d) `double x = 1.1, y = 2.2, z = 3.3;`
e) `double x,y,z = 1.0,2.0,3.0;`

Quiz: Question 21

- Which of the following variable declarations are valid in ANSI-C?
  (Check all that apply!)

a) `double xyz;`
b) `double x, y, z;`
c) `double x = 1.0;`
d) `double x = 1.1, y = 2.2, z = 3.3;`
e) `double x,y,z = 1.0,2.0,3.0;`

Quiz: Question 22

- Which of the following data types has the largest range of representable numbers?
  a) `char`
  b) `short int`
  c) `long long int`
  d) `unsigned int`
  e) `signed long int`

Quiz: Question 22

- Which of the following data types has the largest range of representable numbers?
  a) `char`
  b) `short int`
  c) `long long int`
  d) `unsigned int`
  e) `signed long int`

Quiz: Question 23

- Which of the following data types can store the greatest value?
  a) `long int`
  b) `long long int`
  c) `unsigned long long int`
  d) `float`
  e) `double`

Quiz: Question 23

- Which of the following data types can store the greatest value?
  a) `long int`
  b) `long long int`
  c) `unsigned long long int`
  d) `float`
  e) `double`

Quiz: Question 24

- In the **gdb** debugger, what does **`next`** do?

a) It moves to the next argument of the function.
b) It calls the next function in the program.
c) It executes the next statement in the program.
d) It prints the value of the next variable.
e) It loads the next program into the debugger.

Quiz: Question 24

- In the **gdb** debugger, what does **`next`** do?

a) It moves to the next argument of the function.
b) It calls the next function in the program.
c) It executes the next statement in the program.
d) It prints the value of the next variable.
e) It loads the next program into the debugger.

Quiz: Question 25

- Assume that **x** is an integer in the range of 1 through 10 inclusively.
  Which of the following expressions can be used as a test for **x** being an even number?
  (Check all that apply!)

a) `x % 2 == 0`
b) `x / 2 > 1`
c) `x % 2 == 1`
d) `x / 2 * 2 == x`
e) `x==2 || x==4 || x==6 || x==8 || x==10`

Quiz: Question 25

- Assume that **x** is an integer in the range of 1 through 10 inclusively.
  Which of the following expressions can be used as a test for **x** being an even number?
  (Check all that apply!)

a) `x % 2 == 0`
b) `x / 2 > 1`
c) `x % 2 == 1`
d) `x / 2 * 2 == x`
e) `x==2 || x==4 || x==6 || x==8 || x==10`

Quiz: Question 26

- Given the following function **grade**, what is the result of **grade(85)**?

a) **'A'**
b) **'B'**
c) **'C'**
d) **'D'**
e) **'F'**

```
char grade(int n)
{ char g = 'x';
  switch(n/10)
  { case 10:
    case  9: g = 'A';
    case  8: g = 'B';
    case  7: g = 'C';
    case  6: g = 'D';
    default: g = 'F';
   }
  return g;
}
```

EECS22: Advanced C Programming, Lecture 13 (c) 2017 R. Doemer 53

Quiz: Question 26

- Given the following function **grade**, what is the result of **grade(85)**?

a) **'A'**
b) **'B'**
c) **'C'**
d) **'D'**
e) **'F'**

```
char grade(int n)
{ char g = 'x';
  switch(n/10)
  { case 10:
    case  9: g = 'A';
    case  8: g = 'B';
    case  7: g = 'C';
    case  6: g = 'D';
    default: g = 'F';
   }
  return g;
}
```

EECS22: Advanced C Programming, Lecture 13 (c) 2017 R. Doemer 54

Quiz: Question 27

- What is the value of **x** after the following code fragment is executed?

```
int x = 0;
for(x = 1; x <= 10; x++)
   {  }
```

a) **0**
b) **1**
c) **9**
d) **10**
e) **11**

Quiz: Question 27

- What is the value of **x** after the following code fragment is executed?

```
int x = 0;
for(x = 1; x <= 10; x++)
   {  }
```

a) **0**
b) **1**
c) **9**
d) **10**
e) **11**

Quiz: Question 28

- Given the following program fragment, what is printed when it gets executed?

a) nothing
b) **0**
c) **10**
d) **20**
e) **30**

```
int i = 1;
int s = 0;
while (1)
  { i++;
    if (i >= 10)
       { break; }
    if (i % 2 == 1)
       { continue; }
    s += i;
   }
printf("%d", s);
```


Quiz: Question 28

- Given the following program fragment, what is printed when it gets executed?

a) nothing
b) **0**
c) **10**
d) **20**
e) **30**

```
int i = 1;
int s = 0;
while (1)
  { i++;
    if (i >= 10)
       { break; }
    if (i % 2 == 1)
       { continue; }
    s += i;
   }
printf("%d", s);
```

Quiz: Question 29

- Given the following code fragment, which of the following statements are true? (Check all that apply!)

```
double f(int x);
void g(int x, int y)
{
  int z;

  z = f(x) + 2*y;
  return z;
}
```

a) Function **f** is declared.
b) Function **g** calls function **f**
c) Variable **z** is a local variable of function **g**
d) Function **g** is declared and defined.
e) **y** is a parameter of function **g**.

Quiz: Question 29

- Given the following code fragment, which of the following statements are true? (Check all that apply!)

```
double f(int x);
void g(int x, int y)
{
  int z;

  z = f(x) + 2*y;
  return z;
}
```

a) Function **f** is declared.
b) Function **g** calls function **f**
c) Variable **z** is a local variable of function **g**
d) Function **g** is declared and defined.
e) **y** is a parameter of function **g**.

Quiz: Question 30

- Given the following program fragment, what is the value of **`g(2,f(3,4))`**?

a) **8**
b) **9**
c) **10**
d) **11**
e) **12**

```
int x = 7;

int f(int x, int y)
{
  return x + y;
}

int g(int x, int y)
{
  return f(y, x);
}
```


Quiz: Question 30

- Given the following program fragment, what is the value of **`g(2,f(3,4))`**?

a) **8**
b) **9**
c) **10**
d) **11**
e) **12**

```
int x = 7;

int f(int x, int y)
{
  return x + y;
}

int g(int x, int y)
{
  return f(y, x);
}
```