EASTERN WASHINGTON UNIVERSITY
start something big

C Looping & I/O

Computer Science Department
Eastern Washington University
Yun Tian (Tony) Ph.D.

Last Class

- If
- Switch
- Basic I/O

Topic for today

- More Basic I/O with I/O redirection
- Looping has same usage as in Java
  - While
  - For
  - Do while

Demo of Formatted Output

```
#include <stdio.h>
#include <math.h>
int main()
{
   int year = 1990;
   double rate = 1.2f;
   double amount = 1000.0f;
   int i;

   printf( "%4s%21s\n", "Year", "Amount on Year" );
   for( i = 0; i < 10; i ++ )
   {
      amount = amount * pow(rate, i);
      printf( "%4d%21.2f\n", year ++, amount );
   }
}
```

pow() function

- The function pow(x, y) calculates the value of x raised to the **y**th power.
- It takes two arguments of type double and returns a double value.
- This program would malfunction without the inclusion of math.h,
  - as the linker would be unable to find the pow function.

pow() function

- Function pow requires two double arguments, but variable year is an integer.
- The math.h file includes information that tells the compiler to convert the value of year to a temporary double representation before calling the function.

Formatting Output

- printf("%4d%21.2f\n", year ++, amount);
- The conversion specifier %21.2f is used to print the value of the variable amount in the program.
- The **21** in the conversion specifier denotes the field width in which the value will be printed.
  - A field width of 21 specifies that the value printed will appear in 21 print positions.
- The 2 specifies the precision (i.e., the number of decimal positions).

Formatting Output

- If the number of characters displayed is less than the field width, then the value will automatically be **right justified** in the field.
  - printf("%7.2f", a); // value of a is 3.14159

			3	.	1	4

- This is particularly useful for aligning floating-point values with the same precision (so that their decimal points align vertically).

Formatting Output

- To left justify a value in a field, place a - (minus sign) between the % and the field width.
  - printf("%-7.2f", a); // value of a is 3.14159

3	.	1	4			

- The minus sign may also be used to left justify integers (such as in %-6d) and character strings (such as in %-8s).

char and int

- Characters are normally stored in variables of type char.
- However, an important feature of C is that characters can be stored in any integer data type
  - because they're usually represented as one-byte integers in the computer.

char and int

- Thus, we can treat a character as either an integer or a character, depending on its use.

  E.g. printf("The character (%c) has the value %d.\n", 'a', 'a');
- It uses the conversion specifiers %c and %d to print the character **a** and its integer value, respectively.
- The result is: The character (a) has the value 97.
- The integer 97 is the character's numerical representation(ASCII code) in the computer.

Demo of Looping

- Diamond printing

- PI approximation
  - Calculate the value of pi from the infinite series.

$$\pi = 4 - \frac{4}{3} + \frac{4}{5} - \frac{4}{7} + \frac{4}{9} - \frac{4}{11} + \cdots$$

Demo of Looping

- I/O redirection with C program
  - redirect.c
  - studentGrades.txt

Demo of Looping

- How to read in a text file with I/O redirection in C program?
  - Write your program as if you input from the keyboard.
    - The input format should match the format of the text file you like to read.
  - Compile your program into executable, e.g. **myApp**
  - Run your program using ./myApp < fileToRead.txt

Summary

- Looping
- Formatting output
- Char and int
- I/O redirection with C program