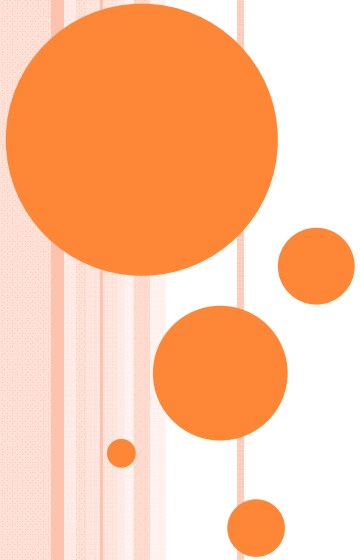


Summer Session I 2014
EECS 10 Week3 Discussion2

Che-Wei Chang

OUTLINE

- Reminder: 2nd midterm is coming.
- Assignment 3 Part 2
  - Menu Driven Calculator for floating point number [30pts]
- Concept review: Function

ASSIGNMENT 3

- Calculator
  - Deadline : 07/14/2014
- Name your files calculator.c, calculator.txt, and calculator.script
- Make sure your program is free of warning
  - Use –Wall option to show all warnings.
- Hints
  - Before you implement your work, review lecture slides about function declaration, function definition, and function call.
  - Read the assignment handout carefully
- Menu driven calculator for floating point number
  - Good exercise for function call
  - What is the input? What is the output?
  - What is the control flow for this program?
  - How to implement this program?

Menu Driven Calculator

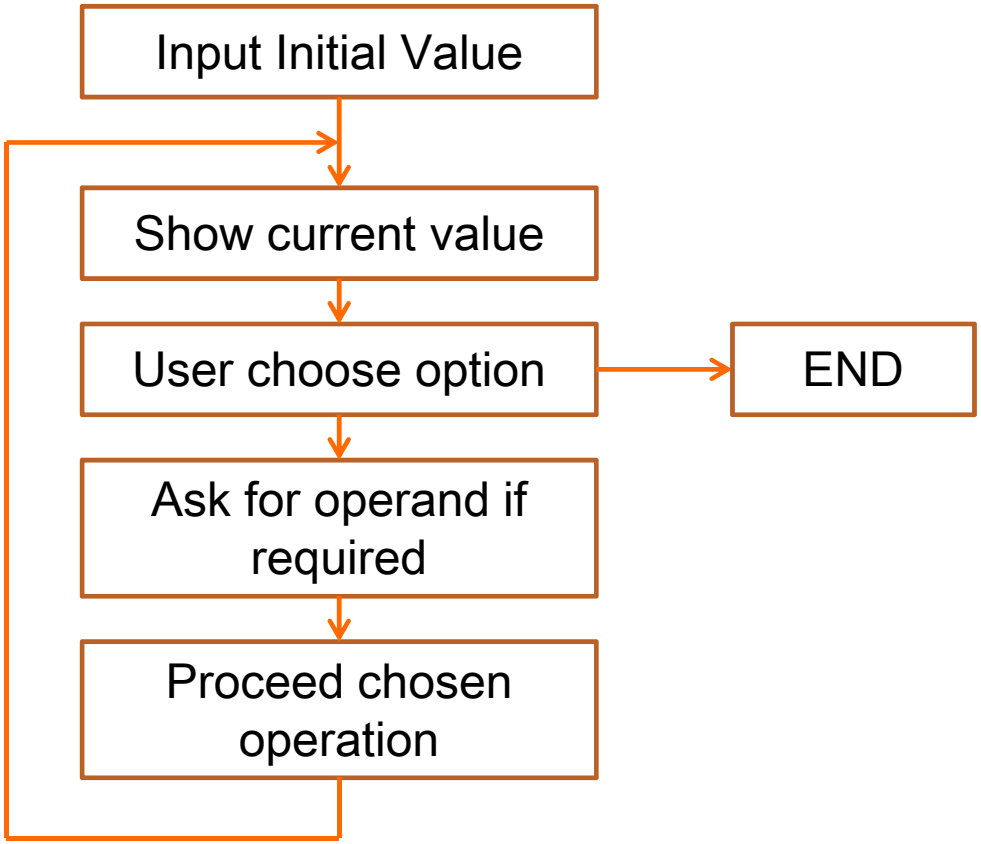
- Prompt a menu and user can choose the operation
- Operation List
  - 1. Add
  - 2. Subtract
  - 3. Multiply
  - 4. Divide
  - 5. Absolute
  - 6. Quit

MENU DRIVEN CALCULATOR

- Should be aware of the following points:
  - The execution only finishes when the user want to quit
    - Infinite loop, with breaking condition
  - Asking for additional operand if required
    - Ex: user chooses addition → asking for the 2nd operand
    - Ex: user chooses absolute → 2nd operand is not needed.
  - A function has to be created for the corresponding operation
    - Ex:
```
/*function Add for add operation*/
double Add(double op1, double op2)
```
  - Misuse of the calculator must be handled
    - Notify the user with error message when the input is invalid
    - Ex: report "ERROR: Division by zero" when user input 0 as the divisor.

Menu Driven Calculator

- Flow:



Concept review: Function

- Important programming concepts
- C programming language distinguishes 3 constructs around functions:
  - Function declaration
    - Declaration of function name, parameters, and return type.
  - Function definition
    - An implicit function declaration
    - Extension of a function declaration with a function body
    - Function declaration + function behavior
  - Function call
    - Invocation of a function
    - Supply argument for formal parameters

C PROGRAM RULES

- Rules for function
  - A function must be declared before it can be called
  - Multiple function declarations are allowed (if they match)
  - A function declaration is an implicit function declaration
  - A function must be defined exactly once in a program
  - A function can be called for any number of times
- Scope of an identifier
  - Portion of the program where the identifier can be referred
  - Global variables *file scope*
  - Function parameter *function scope*
  - Local parameter *block scope*

FUNCTION –

USE CALCULATOR AS THE EXAMPLE

```
#include <stdio.h>
/*global variable declaration*/
double currValue ;
double newValue ;
…
/*function declaration*/
void getInput() ;
double Add (double, double) ;
…
/*main function*/
int main (void)
{…}
/*function definition*/
void getInput()
{…}
double Add (double op1, double op2)
{…}
…
```

SCOPE – GLOBAL FUNCTION

```
#include <stdio.h>
/*global variable declaration*/
double currValue ;
double newValue ;
…
/*function declaration*/
void getInput() ;
double Add (double, double) ;
…
/*main function*/
int main (void)
{…}
/*function definition*/
void getInput()
{…}
double Add (double op1, double op2)
{…}
…
```

Scope of global function
printf(), scanf(), etc.

SCOPE – GLOBAL VARIABLE

```
#include <stdio.h>
/*global variable declaration*/
double currValue ;
double newValue ;
…
/*function declaration*/
void getInput() ;
double Add (double, double) ;
…
/*main function*/
int main (void)
{…}
/*function definition*/
void getInput()
{…}
double Add (double op1, double op2)
{…}
…
```

Scope of global variable currValue ;

SCOPE – GLOBAL FUNCTION

```
#include <stdio.h>
/*global variable declaration*/
double currValue ;
double newValue ;
…
/*function declaration*/
void getInput() ;
double Add (double, double) ;
…
/*main function*/
int main (void)
{…}
/*function definition*/
void getInput()
{…}
double Add (double op1, double op2)
{…}
…
```

Scope of global function
Add ;

IMPLICIT FUNCTION DECLARATION

```
#include <stdio.h>
/*global variable declaration*/
double currValue ;
double newValue ;
…
/*function declaration*/
void getInput() ;
double Add (double, double) ;
double Sub (double, double) ;
…
/*main function*/
int main (void)
{…}
/*function definition*/
void getInput()
{…}
double Add (double op1, double op2)
{…}
…
```

IMPLICIT FUNCTION DECLARATION

```
#include <stdio.h>
/*global variable declaration*/
double currValue ;
double newValue ;
…
/*function definition*/
void getInput()
{…}
double Add (double op1, double op2)
{…}
…
/*main function*/
int main (void)
{…}
```

Scope of global function
Add ;

IMPLICIT FUNCTION DECLARATION

```
#include <stdio.h>
/*global variable declaration*/
double currValue ;
double newValue ;

/*main function*/
int main (void)
{…}
/*function definition*/
void getInput()
{…}
double Add (double op1, double op2)
{…}
…
```

Compiler will complain

Scope of global function Add ;

SCOPE – LOCAL PARAMETER

```
#include <stdio.h>
/*global variable declaration*/
double currValue ;
double newValue ;
…
/*function definition*/
void getInput()
{…}
double Add (double op1, double op2)
{
  double sum ;
  sum = op1 + op2 ;
  return sum ;
}
/*main function*/
int main (void)
{…}
```

Scope of parameter op1 ;

SCOPE – LOCAL VARIABLE

```
#include <stdio.h>
/*global variable declaration*/
double currValue ;
double newValue ;
…
/*function definition*/
void getInput()
{…}
double Add (double op1, double op2)
{
  double sum ;
  sum = op1 + op2 ;
  return sum ;
}
/*main function*/
int main (void)
{…}
```

Scope of parameter sum ;