

Function Programming Errors

- **Syntax errors**
 - May be detected by compiler
- **Usage errors**
 - Undetected, but program fails
- **Design errors**
 - May need some guidelines

Syntax error

Missing Prototype

```

int Roundoff (double arg);
void main ()
{
  int x = RoundOff (3.5);
}
int Roundoff (double arg)
{
  int intval = int(arg + 0.5);
  return intval;
}

```

Syntax error

Prototype/Definition Mismatch

```

double Roundoff (double arg);
void main ()
{
  double x = RoundOff (3.5);
}
int Roundoff (double arg)
{
  int intval = int(arg + 0.5);
  return intval;
}

```

```

:
:
: Syntax error
Extra Semicolon in Definition
int Roundoff (double arg);

void main ()           What happens?
{
  int x = RoundOff (3.5);
}

int Roundoff (double arg) ;
{
  int intval = int(arg + 0.5);
  return intval;
}

```

```

:
:
: Syntax error
Argument Mismatch
int Sum (int a, int b);

void main ()           What happens?
{
  int x = Sum (-7,4,35);
}

int Sum (int a, int b)
{
  int ret_val = a + b;
  return ret_val;
}

```

```

:
:
: Syntax error
Procedure as Operand
void Out(int value);

void main ()           What happens?
{
  int x = 2 + Out(-53);
}

void Out(int value)
{
  cout << value << endl;
}

```

Syntax error

Missing Parentheses

```

void Prompt();
void main ()
{
  int x;
  Prompt;
  cin >> x;
}

void Prompt()
{
  cout << "Enter x: " << endl;
}

```

May give no error, but just not work!

Syntax error

Illegal Reference to Local Object

```

void Print();
void main ()
{
  int x;
  Print();
}

void Print()
{
  cout << x << endl;
}

```

Scope of x is body of main; not visible inside Print

Usage error

Value vs Reference Argument (1)

```

void Increase(int x);
void main ()
{
  int x = 3;
  Increase(x);
  cout << x << endl;
}

void Increase(int x)
{
  x = x + 10;
}

```

Should be "int& x"

To modify actual argument, must be pass-by-reference.

Usage error

Value vs Reference Argument (2)

```

void Print(int& x);
void main ()
{
  int x = 3;
  Print(x);
}

void Print(int& x)
{
  cout << x << endl;
}

```

Should be "int x"

If not modifying actual argument,
normally use pass-by-value.

Usage error

Literal as Reference Argument

```

void Increase(int& x);
void main ()
{
  Increase(12);
}

void Increase(int& x)
{
  x = x + 10;
}

```

Literal cannot be an lvalue, so
illegal as reference argument.

Usage error

Return Too Early

```

void Out(int a, int b);
void main ()
{
  Out(15, 27);
}

void Out(int a, int b)
{
  cout << a;
  return;
  cout << b << endl;
}

```

Statement after return not executed.

Design Errors

- **Unneeded global objects**
 - Use arguments to pass data instead
- **Functions that are "too long"**
 - Hard to comprehend (even `main()`)
- **Too many arguments**
- **Failing to document function**

Refining OO Designs

- **Object implementation**
 - Previously, sections of code in one function (`main`)
 - Now, use functions for some objects
- **Ownership**
 - Package items by “need to know”

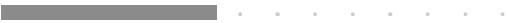
Convenience Store Problem

- **Version comparison**
 - Page 15 versus page 124
- **Objects in functions**
 - Computing: `priceCanadian`
 - Input: `inputInt`
 - Output: `outputDouble`

⋮

Item Ownership

- **Common objects**
 - Input, output values
 - Defined in main
- **Private objects**
 - Conversion constants
 - Defined in priceCanadian



Handwriting lines for notes corresponding to the 'Item Ownership' section.

⋮

Common Objects

- **Object used by more than one function**
- **How should it be accessed?**
 - Global (file scope) object?
 - Passed as function argument?
 - Defined in “parent” routine



Handwriting lines for notes corresponding to the 'Common Objects' section.

⋮

Changemaker Program

- **Text, page 128**
 - Updated from page 21
- **How is it different?**
 - How is the problem decomposed?
- **Can you do better?**
 - What about those eight functions?



Handwriting lines for notes corresponding to the 'Changemaker Program' section.