

Computers in Engineering COMP 208

Expressions
Michael A. Hawker

The Speed of Light

- How long does it take light to travel from the sun to earth?
- Light travels 9.46×10^{12} km a year
- A year is 365 days, 5 hours, 48 minutes and 45.9747 seconds long
- The average distance between the earth and sun is 150,000,000 km

Sept. 13th, 2007

Expressions

2

Elapsed Time

```
PROGRAM light_travel
  IMPLICIT NONE
  REAL :: light_minute, distance, time
  REAL :: light_year = 9.46 * 10.0 ** 12

  light_minute = light_year / (365.25 * 24.0 * 60.0)
  distance = 150.0 * 10.0 ** 6
  time = distance / light_minute

  WRITE (*,*) "Light from the sun takes ", time, &
    "minutes to reach earth."

END PROGRAM light_travel
```

Sept. 13th, 2007

Expressions

3

Arithmetic Expressions

- An arithmetic expression is formed using the operations:
 - + (addition)
 - (subtraction)
 - * (multiplication),
 - / (division)
 - ** (exponentiation)

Sept. 13th, 2007 Expressions 4

Watch out for Ambiguity

- Let's look at two expressions from our program
 - `light_minute = light_year / (365.25 * 24.0 * 60.0)`
 - `distance = 150.0 * 10.0 ** 6`
- What value is assigned to distance?
 - $(150.0 * 10.0) ** 6$?
 - $150.0 * (10.0 ** 6)$?

Sept. 13th, 2007 Expressions 5

Watch out for ambiguity

- Let's look at an expression from our program
 - `light_minute = light_year / (365.25 * 24.0 * 60.0)`
- What if the expression didn't have parentheses?
 - `light_minute = light_year / 365.25 * 24.0 * 60.0`

Sept. 13th, 2007 Expressions 6

Watch out for ambiguity

- How about another expression?

```
distance = 150.0 * 10.0 ** 6
```

- What value is assigned to distance?

```
(150.0 * 10.0) ** 6 ?
```

```
150.0 * (10.0 ** 6) ?
```

Sept. 13th, 2007

Expressions

7

Precedence Rules

- Every language has rules to determine what order to perform operations
- These rules try to mimic the conventions we learn growing up
- For example, in FORTRAN ** comes before *
- In an expression, all of the **'s are evaluated before the *'s

Sept. 13th, 2007

Expressions

8

Precedence Rules

- First evaluate operators of higher precedence

- $3 * 4 - 5 \rightarrow ?$

- $3 + 4 * 5 \rightarrow ?$

Sept. 13th, 2007

Expressions

9

Precedence Rules

- First evaluate operators of higher precedence
 - $3 * 4 - 5 \rightarrow 7$
 - $3 + 4 * 5 \rightarrow 23$

Sept. 13th, 2007 Expressions 10

Precedence Rules

- For operators of the same precedence, use associativity. Exponentiation is right associative, all others are left associative
 - $5 - 4 - 2 \rightarrow ?$
 - $2 ** 3 ** 2 \rightarrow ?$

Sept. 13th, 2007 Expressions 11

Precedence Rules

- For operators of the same precedence, use associativity. Exponentiation is right associative, all others are left associative
 - $5 - 4 - 2 \rightarrow -1$ (not 3)
 - $2 ** 3 ** 2 \rightarrow 512$ (not 64)
- Expressions in Parenthesis are Evaluated First
 - $(5 - 3) * 4 \rightarrow 8$

Sept. 13th, 2007 Expressions 12

Precedence of Operators in FORTRAN

- Operators in order of precedence and their associativity:

Arithmetic	
**	right to left
*, /	left to right
+, -	left to right
Relational	
<, <=, >, >=, ==, /=	no associativity
Logical	
.NOT.	right to left
.AND.	left to right
.OR.	left to right
.EQV., .NEQV.	left to right

Sept. 13th, 2007 Expressions 13

Another Example

- Last lecture we looked at the problem of finding the roots of a quadratic equation
- We focused on the discriminant
- Here is a program that computes the roots

Sept. 13th, 2007 Expressions 14

```

! Solve Ax^2 + Bx + C = 0
! -----
PROGRAM QuadraticEquation
  IMPLICIT NONE

  REAL :: a, b, c
  REAL :: d
  REAL :: root1, root2

! read in the coefficients a, b and c
  WRITE(*,*) 'A, B, C Please : '
  READ(*,*) a, b, c

! compute the square root of discriminant d
  d = SQRT(b*b - 4.0*a*c)

! solve the equation
  root1 = (-b + d)/(2.0*a) ! first root
  root2 = (-b - d)/(2.0*a) ! second root

! display the results
  WRITE(*,*) 'Roots are ', root1, ' and ', root2
END PROGRAM QuadraticEquation

```

Sept. 13th, 2007 Expressions 15

Data Types

- In the examples we have declared the variables to be of type REAL
- That is, each memory cell can hold a real number
- What is a real number?
 - In Mathematics?
 - In FORTRAN?

Sept. 13th, 2007 Expressions 16

Real Numbers (examples)

3.14159
10.0
10.
-123.45
+1.0E-3 (0.001)
150.0E6

But Not:

1,000.000
123E5
12.0E1.5

Sept. 13th, 2007 Expressions 17

Real Numbers (representation)

- A real value is stored in two parts
 - A mantissa determines the precision
 - An exponent determines the range
- Real numbers are typically stored as either
 - 32 bits (4 bytes): type REAL
 - 64 bits (8 bytes): type DOUBLE
- (This varies on some computers)

Sept. 13th, 2007 Expressions 18

Accuracy of Real Numbers

* REAL numbers:

- Mantissa represented by 24 bits gives about 7 decimal digits of precision
- Exponent represented by 8 bits gives range from 10^{-38} to 10^{38}

* DOUBLE numbers:

- Mantissa represented by 53 bits gives about 15 decimal digits of precision
- Exponent represented by 11 bits gives range from 10^{-308} to 10^{308}

Sept. 13th, 2007

Expressions

19

Rounding Errors

* Be careful not to expect exact results with real numbers

```
PROGRAM roundoff
  IMPLICIT NONE
  REAL :: x, y
  x = 100.00002
  y = x*x - x
  WRITE (*,*) x/y * (x - 1)
END PROGRAM roundoff
```

Sept. 13th, 2007

Expressions

20

2.0 + 2.0 = ???

* What result do we expect?

$$\frac{x}{x^2 - x} \times (x - 1)$$

* What result do we get?

```
>gfortran -fimplicit-none -W -Wall
    "roundoff.f90" -o "roundoff.exe"
>Exit code: 0
>roundoff
0.99999999
>Exit code: 0
```

Sept. 13th, 2007

Expressions

21

Back to The Speed of Light

- How long does it take light to travel from the sun to earth?
- The result we got was a decimal number of minutes
- We'd rather have the number of minutes and seconds

Sept. 13th, 2007

Expressions

22

Minutes and Seconds

```
PROGRAM light_travel
  IMPLICIT NONE
  REAL :: light_minute, distance, time
  REAL :: light_year = 9.46 * 10.0**12
  INTEGER :: minutes, seconds

  light_minute = light_year / (365.25 * 24.0 * 60.0)
  distance = 150.0 * 10**6
  time = distance / light_minute

  minutes = time
  seconds = (time - minutes) * 60

  WRITE (*,*) "Light from the sun takes ", minutes, &
    " minutes and ", seconds, " seconds to reach earth."

END PROGRAM light_travel
```

Sept. 13th, 2007

Expressions

23

Integer Numbers

- Integers are whole numbers represented using 32 (or 16 or 64 bits)
- For 32 bit numbers whole numbers between -2147483648 and 2147483647 can be represented

0
-987
+17
123456789

Sept. 13th, 2007

Expressions

24

Integer Arithmetic

- The result of performing an operation on two integers is an integer
- This may result in some unexpected results since the decimal part is truncated

$12/4 \rightarrow ?$
 $13/4 \rightarrow ?$
 $1/2 \rightarrow ?$
 $2/3 \rightarrow ?$

Sept. 13th, 2007 Expressions 25

Integer Arithmetic

- The result of performing an operation on two integers is an integer
- This may result in some unexpected results since the decimal part is truncated

$12/4 \rightarrow 3$
 $13/4 \rightarrow 3$
 $1/2 \rightarrow 0$
 $2/3 \rightarrow 0$

Sept. 13th, 2007 Expressions 26

Some Simple Examples

$2 + 2 \rightarrow ?$
 $1.25 - 0.45 \rightarrow ?$
 $4 * 8 \rightarrow ?$
 $7.4/1.25 \rightarrow ?$
 $9.6/4.8 \rightarrow ?$
 $-5 * 2 \rightarrow ?$
 $3/5 \rightarrow ?$
 $3./5. \rightarrow ?$
 $(4 + 8) * 2 / 3 \rightarrow ?$

Sept. 13th, 2007 Expressions 27

Another Example

```
2 * 4 * 5 / 3 ** 2
--> 2 * 4 * 5 / [3 ** 2]
--> 2 * 4 * 5 / 9
--> [2 * 4] * 5 / 9
--> 8 * 5 / 9
--> [8 * 5] / 9
--> 40 / 9
--> 4
```

The result is 4 rather than 4.444444 since the operands are all integers.

Sept. 13th, 2007

Expressions

28

Mixed Mode Assignment

- In the speed of light example, we assigned a real value to an integer variable

```
minutes = time
```

- The value being assigned is converted to an integer by truncating (not rounding)
- When assigning an integer to a real variable, the integer is first converted to a real (the internal representation changes)

Sept. 13th, 2007

Expressions

29

Mixed Mode Expressions

- In the speed of light example, we subtracted the integer value, minutes, from the real value, time

```
seconds = (time - minutes) * 60
```

- If an operation involves an integer and a real, the integer is first converted to a real and then the operation is done.
- The result is real.
- The example has two arithmetic operations, an assignment and forces two type conversions

Sept. 13th, 2007

Expressions

30

Mixed Mode Examples

```
1 + 2.5 → 3.5
1/2.0 → 0.5
2.0/8 → 0.25
-3**2.0 → -9.0
4.0**(1/2) → 1.0 (since 1/2 → 0)
```

Sept. 13th, 2007

Expressions

31

Another Example

```
25.0 ** 1 / 2 * 3.5 ** (1 / 3)
→ [25.0 ** 1] / 2 * 3.5 ** (1 / 3)
→ 25.0 / 2 * 3.5 ** (1 / 3)
→ 25.0 / 2 * 3.5 ** ([1 / 3])
→ 25.0 / 2 * 3.5 ** 0
→ 25.0 / 2 * [3.5 ** 0]
→ 25.0 / 2 * 1.0
→ [25.0 / 2] * 1.0
→ 12.5 * 1.0
→ 12.5
```

Sept. 13th, 2007

Expressions

32

Something's Not Right Here

```
PROGRAM light_travel
  IMPLICIT NONE
  REAL :: light_minute, distance, time
  REAL :: light_year = 9.46 * 10**12

  light_minute = light_year / (365.25 * 24.0 * 60.0)
  distance = 150.0 * 10**6
  time = distance / light_minute

  WRITE (*,*) "Light from the sun takes ", time, &
    "minutes to reach earth."

END PROGRAM light_travel
```

Sept. 13th, 2007

Expressions

33

What Happened?

- Look at this assignment:
`light_year = 9.46 * 10**12`
- Precedent rules tell us that `10**12` is evaluated first
- Type rules tell us that the result is an integer
- Integers can only have about 9 digits, not 12

Sept. 13th, 2007 Expressions 34

How do we fix it?

- Let's change
`light_year = 9.46 * 10**12`
- to
`light_year = 9.46 * 10.0**12`
- That works, but why?
 - REAL numbers can represent much larger exponents

Sept. 13th, 2007 Expressions 35

Where Are We

- We have seen
 - Expressions
 - Types and declarations
- Coming up
 - Selection Mechanisms
 - Looping

Sept. 13th, 2007 Expressions 36
