

---

# CS 241

## Control Structures

Christopher A. Gantz

SPS Undergraduate Program  
Regis University  
cgantz@regis.edu

---

## Lecture 2b

### Overview of CS 241: Data Types And Output

Christopher A. Gantz  
School of Professional Studies  
Regis University  
cgantz@regis.edu

# Overview

---

- Reading
  - Nance textbook Pages 40-51
- Data Types and Output
  - Pascal Data Types
  - Integers, Reals, Chars, and strings
  - Basic Output
  - Examples

# Pascal Data Types

---

- A data type is a formal description of the set of values a type name defines
- Pascal requires that all data represented by both variables and constants either be declared with or have an implied data type
-

# Pascal Data Types

---

- Pascal has the following 4 basic data types
  - integer
  - real
  - char
  - string

# The Integer Data Type

---

- The integer data type represents numbers that are positive, negative, or zero
- Integer type rules
  - (+) doesn't have to precede a positive integer, E.g. +512 and 512 are the same
  - (-) signs must be utilized for negative integers
  - Leading zero's are ignored
  - Decimal points cannot be utilized when defining integers
  - Commas cannot be utilized when writing integers

# The Integer Data Type (Cont)

---

- The integer data type has a limit on both the largest and smallest integer constant that can be represented
  - Maxint – largest integer value
  - -Maxint – smallest integer value
- The above standard integer constants are defined and recognized by every version of Pascal
- Note: Different machine architectures have different constant values for these constants

# The Integer Data Type (Cont)

---

- Example program
-

# The Real data type

---

- Real numbers in pascal are expressed through the use of decimal notation
- Decimal notation requires the use of a decimal point with at least one digit on each side of the decimal (E.g. 0.03 and NOT .3)
- The (+) and (-) signs work for real data type variables and constants in the same manner as integers

# The Real data type (Cont.)

---

- Real data type values can be expressed in either
  - Fixed point notation
  - Floating point notation
- Floating point notation is also referred to as exponential form or scientific notation
  - E.g. 0.00075     $7.5 \times 10^{-4}$     7.5E-4
- See Table 2.4 on pg 41 of text for more examples

# The char data type

---

- Data type utilized to represent character variables and constants
- The char data type can represent only one single character
- These characters originate from the ASCII character set which includes
  - Letters of the alphabet
  - The digits 0, 1, 2, ..., 9
  - Special character symbols E.g. #, &, !, etc.

# The string data type

---

- A string constant is a finite collection of one or more characters
- String constants are usually defined in the constant section of a pascal program
- String constants **MUST** be enclosed within single quotation marks
  - E.g. 'This is a string constant!'
- The string data type is used to declare variables of type string

# Basic Output

---

- The basic output construct is used to print information to a display, printer, or file
- Pascal provides the following 2 output statements
  - Write
  - Writeln (pronounced “write line”)
- BNF form
  - write(<expression 1>, <expression 2>, ..., <expression n>)
  - writeln(<expression 1>, <expression 2>, ..., <expression n>)

# Output Examples

---

- `Write('Please enter a number? ');`
- `Writeln('This is an output example!');`

# Output Formatting

---

- Output formatting allows the specification of a field width when printing the basic data types of Pascal
- For integer data type expressions
  - `write(<expression>:<n>)`
    - E.g. `write(123:5)`
  - `writeln(<expression>:<n>)`
- The integer printed will be right justified in the specified field

# Output Formatting

---

- For real data type expressions
  - `write(<expression>:<n>:<n>)`
  - `writeln(<expression>:<n>:<n>)`
    - E.g. `writeln(3.1457:2:3)`
- If no formatting is used , the output of a real will be in floating point form

# Output Formatting (Cont)

---

- For string data type expressions
  - `write(<expression>:<n>)`
    - E.g. `write('This is a test':15)`
  - `writeln(<expression>:<n>)`
    - E.g. `writeln('This is another test':10)`
- The string expression will be right justified in the field and strings are truncated when necessary

# Construct Summary

---