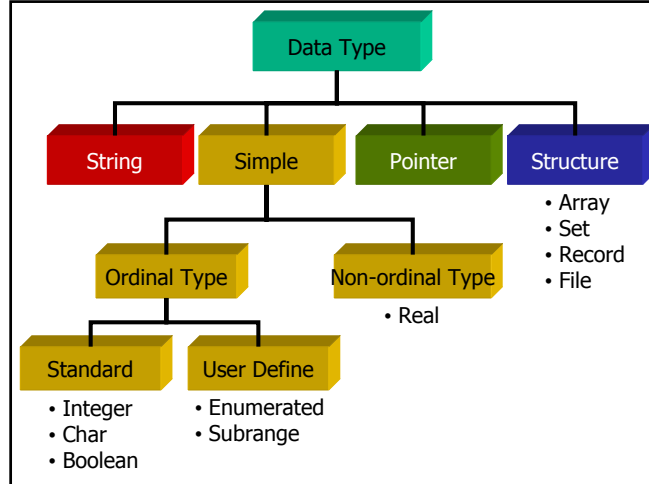


Data Type

Anan Phonphoem
anan@cpe.ku.ac.th



Why need both Integer & Real?

- Integer operations are faster
- Integer needs less storage space
- Operations with integers are precise (Real may loss accuracy)

Integer & Real

4
 Binary number

- Integer Format**
- binary number
 - 00100

4.0
 Mantissa Exponent

- Real Format**
- Mantissa (binary fraction between [0.5,1.0] and [-0.5,-1.0])
 - Exponent (power of 2)
 - Real number = mantissa * 2^{exponent} = 1.0 * 2²

Integer Types

- Shortint -128..127
- Integer -32,768..32,767
- Longint -2,147,483,648.. 2,147,483,647
- Byte 0..255 (unsigned)
- Word 0..65535 (unsigned)

```

Num := 32767; writeln(Num);
Num := Num+1; writeln(Num);
After running
32767
-32768
    
```

Real Types

Type	Range	Digits	Bytes
real	2.9e-39..1.7e38	11-12	6
single	1.5e-45..3.4e38	7-8	4
double	5.0e-324..1.7e308	15-16	8
extended	3.4e-4932..1.1e4932	19-20	10
comp	-9.2e18..9.2e18	19-20	8

Ordinal Types (เลขลำดับ)

- **Integer, Boolean, and char** are ordinal types (**real** is not)
- Each value has a unique predecessor
 - Predecessor of 5 = 4
- Each value has a unique successor
 - Successor of 8 = 9

7

How the ordinality is defined?

- Integer : the value itself
- Non-integer (Boolean, enumerate):
 - First value = 0
 - The next = 1, and so on.
- For char:
 - Ascii code (A=65,B=66,...)

8

The ordinal Functions: Ord, Pred, and Succ

Data Type	Parameter	Ord	Succ	Pred
Integer	15	15	16	14
	0	0	1	-1
	-30	-30	-29	-31
Boolean	False	0	True	Undefined
	True	1	Undefined	False
Char	C	67	D	E
	Blank	32	!	unprintable

9

The ordinal Functions: Ord, Pred, and Succ

Ord

- Ord(-5) → -5
- Ord('A') → 65

Pred

- Pred('B') → A
- Pred(True) → False

Succ

- Succ(459) → 460
- Succ('f') → g

10

Comparison of ordinal-type

- False < True
- 3 < 60
- 'A' < 'S'

11

Char Data Type

- Char can hold exactly one character

```

Program char_input;
Var c1, c2, c3, c4, c5, c6, c7: char;
    i1, i2, i3: integer;
Begin
    read (c1, c2, c3, c4);
    read (i1, i2);
    read (c5, c6, c7);
    read (i3);
End.
    
```

INPUT

Ma#932###14(enter)
876##12(enter)

Result

c1=M, c2=a
C3=#, c4=9
i1=32, i2=14
c5=code 13, c6=code 10
c7=8
i3=76

12

The function chr

- Given a character's ordinality (ASCII code), **chr** returns the corresponding character
- Ord('a') → 97
- Chr(97) → a
- Chr(ord('a')) → a
- 'C'+chr(97)+'ts' → Cats

13

Example

```
For ch:= 'c' to 'g' do  
  writeln(Ord(ch) - Ord('a')+1:3, ch:3);
```

Output ?

14

Practice #1

- Write a program to convert Uppercase to Lowercase letter

Begin

```
if ((ch >='A') and (ch <='Z')) then  
  LowerCase := Chr(Ord(ch) - Ord('A') + Ord('a'))  
else  
  Lowercase := ch;
```

End.

15

Practice #2

- Write a program to display ascii character and its value from 0 to 255 on screen. Each line display 5 characters as followed.

```
...  
66=B 67=C 68=D 69=E 70=F  
71=G 72=H 73=I 74=J 75=K  
...
```

16

Data Type

String

Simple

Pointer

Structure

- Array
- Set
- Record
- File

Ordinal Type

Non-ordinal Type

- Real

Standard

User Define

- Integer
- Char
- Boolean

- Enumerated
- Subrange

Type Declaration

- In Declaration part
- Need to come before using it

```
program Sample(input,output);  
type Number = integer;  
var Grade: Number;  
Begin
```

...

18

Enumerated User Define Type

- Define a name to each value
- For **more understandable program**
- Example:
 - Color = (Blue, Green, Red)
 - StudentYear =
(Freshmen, Sophomore, Junior, Senior)
 - Boolean = (True, False)

19

Enumerated example

```
Program Dress(input, output);
```

```
Type
```

```
Color = (Blue, Green, Red);
```

```
Course = (E204111, E204325);
```

```
Var
```

```
Tie : Color;
```

```
course_teach : Course;
```

```
Begin
```

```
....
```

20

Enumerated type operation

- Logic compare
 - If Course_teach >= E204111 then ...
 - Case Course_teach of
 - E204111: Tie = Blue;
 - E204325: Tie = Green;
- Ordinality
 - Ord(Red) → 2
 - Succ(Green) → Red

21

Enumerated type

- Cannot use with read / write
 - Readln(Tie); writeln(course_teach);

→ Indirect use

```
Writeln('0: Blue');
```

```
Writeln('1: Green');
```

```
Writeln('2: Red');
```

```
Write('Please input tie color'); Readln(num);
```

```
Case num of
```

```
1: Tie = Blue;
```

```
2: Tie = Green;
```

```
3: Tie = Red;
```

```
End;
```

22

Subrange Type

- Define a name for start ...end value
- For **more understandable program**
- Example:

```
var
```

```
Letter: 'a'..'z';
```

```
code: '0'..'9';
```

23

Type Compatibility

- Real \leftrightarrow Integer
 - 4.5 > 2
 - Num: real; num := 2 ...OK
 - Num: integer; num := 2.0 ...Not OK
- Char \leftrightarrow Subrange

```
Type
```

```
upLetter = 'A'..'Z';
```

```
lowLetter = 'a'..'z';
```

```
Var letter1:upLetter; letter2:lowLetter; C:char;
```

```
Begin
```

```
letter1 < letter2 ...OK
```

```
letter2 > C ...OK
```

24

Data Type

String

Simple

Pointer

Structure

- Array
- Set
- Record
- File

Ordinal Type

Non-ordinal Type

- Real

Standard

User Define

- Integer
- Char
- Boolean

- Enumerated
- Subrange