แผนปฏิบัติการสอนสัปดาห์ที่ 5

วิชา 030143361 การโปรแกรมคอมพิวเตอร์สำหรับงานควบคุม **ระดับ:** ปริญญาตรี

เรื่อง ตัวแปรชนิดต่าง ๆ และการแปลงชนิดข้อมูล **เวลา:** บรรยาย 120 นาที
ปฏิบัติ 120 นาที

ก. วัตถุประสงค์การสอน	**รายละเอียดตามที่ระบุไว้ใน**
1. Data Type 2. String และการแปลงข้อมูล 3. Pointer 4. Date-Time 5. ใช้ Function สำเร็จรูปเกี่ยวกับข้อมูล	หน้า 61-80

ข. การจัดการเรียนการสอน

เวลา – นาที		0	60	120	180	240
วัตถุประสงค์		1, 2, 3, 4, 5				
การนำเข้าสู่บทเรียน						
ให้เนื้อหา						
สรุปเนื้อหา						
พัก						
ทำแบบฝึกหัด						
ให้เนื้อหา						
สรุปเนื้อหา						
ทดสอบและเก็บคะแนน						
ประเมินผล		พิจารณาจากผลการทดสอบในชั่วโมงสุดท้าย				
วิธีการสอน:	บรรยาย					
	ถาม – ตอบ					
	ทำแบบฝึกหัด					
	บรรยาย และแสดงให้ดู					
	ทดสอบ					
สื่อการสอน:	คอมพิวเตอร์					
	Presentations					

Tip

- เนื้อหาในบทนี้เป็นการบรรยายในภาคทฤษฎี ให้นศ.จด Note ลงในเอกสารประกอบคำบรรยาย และเขียน Code ทดสอบเพื่อให้เข้าใจในเนื้อหาดียิ่งขึ้น

Content

1. Data type

ในระบบคอมพิวเตอร์มีการแบ่งชนิดข้อมูลออกเป็นหลายชนิด และมีการใช้งานที่แตกต่างกัน ยกตัวอย่างเช่น ข้อมูลที่เป็นตัวหนังสือ Character ข้อมูลที่เป็นตัวเลขจำนวนเต็ม Integer การใช้ข้อมูลให้ถูกต้องเหมาะสมนั้นสำคัญมากถ้ามีการประกาศชนิดข้อมูลที่ผิดจะทำให้เกิด Error ทำให้โปรแกรมไม่สามารถทำงานได้ เช่นประกาศตัวแปรเป็น Integer แต่ต้องการเก็บตัวอักษร 'B' แบบนี้จะเกิด Error ทันที อีกข้อดีของการเลือกใช้ชนิดตัวแปรให้ถูกต้องคือ ความรวดเร็วและกินหน่วยความจำน้อยลง เช่นต้องการเก็บตัวเลข % ของความชื้น เรารู้อยู่แล้วว่ามีค่ามนช่วง 0%-100% ความละเอียด 1% ถ้าเราใช้ Integer จะต้องเสียพื้นที่จัดเก็บ 4 Byte แต่ถ้าเราใช้ Byte จะใช้พื้นที่แค่ 1 Byte ในกรณีที่ค่าความละเอียดต่ำกว่า 1 เช่น 0.1 จะไม่สามารถใช้ข้อมูลจำนวนเต็มได้ต้องใช้ Floating-Point Data ในที่นี้ชนิดข้อมูลที่เหมาะสมที่สุดคือ Single เพราะใช้พื้นที่หน่วยความจำน้อยที่สุด และมีย่านครอบคลุมย่านที่ต้องการใช้งาน ชนิดของข้อมูลในภาษา Pascal มีรายละเอียดดังนี้

Number types

```
// Integer data types :
Int1:Byte;     //                  0 to 255
Int2:ShortInt; //                 -127 to 127
Int3:Word;     //                  0 to 65,535
Int4:SmallInt; //               -32,768 to 32,767
Int5:LongWord; //                  0 to 4,294,967,295
Int6:Cardinal; //                  0 to 4,294,967,295
Int7:LongInt;  //         -2,147,483,648 to 2,147,483,647
Int8:Integer;  //         -2,147,483,648 to 2,147,483,647
Int9:Int64;  // -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807

// Decimal data types :
Dec1:Single;   //  7  significant digits, exponent   -38 to +38
Dec2:Currency; // 50+ significant digits, fixed 4 decimal places
Dec3:Double;   // 15  significant digits, exponent  -308 to +308
Dec4:Extended; // 19  significant digits, exponent -4932 to +4932
```

Tip

- การใช้การเลขฐาน 16 สามารถใช้เครื่องหมาย $ นำหน้า เช่น $3A
- ตัวอย่างการแปลงตัวเลขเป็น Hex String `IntToHex(1234, 1));Ans = 4D2`

Text types

```
Str1:Char;        // Holds a single character, small alphabet
Str2:WideChar;    // Holds a single character, International alphabet
Str3:AnsiChar;    // Holds a single character, small alphabet
Str4:ShortString; // Holds a string of up to 255 Char's
Str5 : String;      // Holds strings of Char's of any size desired
Str6 : AnsiString;  // Holds strings of AnsiChar's any size desired
Str7 : WideString;  // Holds strings of WideChar's of any size desired
```

Logical data types

```
Log1:Boolean;     // Can be 'True' or 'False'
```

Sets, enumerations and subtypes

```
type
  TSuit=(Hearts, Diamonds, Clubs, Spades);    // Defines the enumeration
var
  suit:TSuit;                          // An enumeration variable
```

```
type
  TWeek=Set of 1..7;             // Set comprising the days of the week, by number
var
  week:TWeek;
begin
  week:=[1,2,3,4,5];      // Switch on the first 5 days of the week
end;
```

Assigning to and from variables

```
types
  TWeek=1..7;             // Set comprising the days of the week, by number
```

```
  TSuit = (Hearts, Diamonds, Clubs, Spades);    // Defines an enumeration


const
  FRED='Fred';       // String constant
  YOUNG_AGE=23;          // Integer constant
  TALL:Single=196.9;       // Decimal constant
  NO=False;       // Boolean constant

var
  FirstName, SecondName:String;   // String variables
  Age:Byte;     // Integer variable
  Height:Single;   // Decimal variable
  IsTall:Boolean;  // Boolean variable
  OtherName:String;   // String variable
  Week:TWeek;    // A set variable
  Suit :TSuit;    // An enumeration variable

begin   // Begin starts a block of code statements
  FirstName:=FRED;          // Assign from predefined constant
  SecondName:='Bloggs';      // Assign from a literal constant
  Age:=YOUNG_AGE;     // Assign from predefined constant
  Age:=55;            // Assign from constant - overrides YOUNG_AGE
  Height:=TALL - 5.5;    // Assign from a mix of constants
  IsTall:=NO;           // Assign from predefined constant
  OtherName:=FirstName;     // Assign from another variable
  Week:=[1,2,3,4,5];   // Switch on the first 5 days of the week
  Suit :=Diamonds;      // Assign to an enumerated variable
end;    // End finishes a block of code statements
```

Results

```
FirstName  is now set to 'Fred'
SecondName is now set to 'Bloggs'
Age        is now set to 55
Height     is now set to 191.4
IsTall     is now set to False
OtherName  is now set to 'Fred'
Week       is now set to 1,2,3,4,5
```

```
Suit       is now set to Diamonds  (Notice no quotes)
```

Compound data types

Arrays

```
var
  Suits:array[1..4] of String;    // A list of 4 playing card suit names


begin
  Suits[1]:='Hearts';    // Assigning to array index 1
  Suits[2]:='Diamonds';  // Assigning to array index 2
  Suits[3]:='Clubs';     // Assigning to array index 3
  Suits[4]:='Spades';    // Assigning to array index 4
end;
```

Records

```
type
  TCustomer =Record
    firstName:string[20];
    lastName :string[20];
    age:byte;
  end;
```

```
var
  customer:TCustomer;            // Our customer variable
begin
  customer.firstName:='Fred';    // Assigning to the customer record
  customer.lastName :='Bloggs';
  customer.age :=55;
end;
```

Results

```
customer.firstName is now set to 'Fred'
customer.lastName  is now set to 'Bloggs'
customer.age       is now set to 55
```

Union

```
TPerson = record
 FirstName, LastName: string[40];
 BirthDate: TDate;
 case Citizen: Boolean of
   True: (Birthplace: string[40]);
   False: (Country: string[20];
          EntryPort: string[20];
          EntryDate, ExitDate: TDate);
end;
```

Objects

ส่วนใหญ่จะแปลงจาก Component เป็น Object เมื่อถูกวางลงบน Form หรือจะเป็น Object ที่ถูกสร้างขึ้นตอน Run-Time ก็ได้

Other data types

Files

Pointers

Variants

Type definitions

When we discussed Records above, we introduced the concept of types. Delphi has many predefined data types - both simple, such as string, and compound, such as TPoint (which holds X and Y coordinates of a point). See Type for further details.

http://docwiki.embarcadero.com/RADStudio/Rio/en/Simple_Types_(Delphi)

http://www.delphibasics.co.uk/Article.asp?Name=DataTypes

http://docwiki.embarcadero.com/RADStudio/Rio/en/Delphi_Data_Types_for_API_Integration

2. String และการแปลงข้อมูล

String Class (System.String Class)

Methods

Compare	*Compares two specified String objects.*

CompareTo	Compares the current string with another
Concat	Concatenates one or more instances of String, or the String objects
Copy	Creates a new instance of String with the same value as a specified String.
CopyTo	Copies part of a string into a section of a Char array
EndsWith	Determines if a string ends with a specified string
Equals	Determines whether two String objects have the same value.
Format	Formats numbers, datetimes and enumerations into a string
IndexOf	Finds the first index position of a string within a string
IndexOfAny	Finds the first index of any of a set of characters within a string
Insert	Inserts a string into another string, creating a new, longer string
Join	Converts a string array into a string with a separator string between each
LastIndexOf	Finds the last index position of a string within a string
LastIndexOfAny	Finds the last index of any of a set of characters within a string
PadLeft	Right aligns a string, padding to the left as appropriate
PadRight	Left aligns a string, padding to the right as appropriate
Remove	Removes a number of characters from a string
Replace	Replaces all occurences of a string with another string
Split	Parses a string into an array of substrings
StartsWith	Determines if a string starts with a specified string
SubString	Returns a section of a string
ToCharArray	Creates a character array from a string
ToLower	Converts a string to lower case
ToUpper	Converts a string to upper case
Trim	Trims specified characters from both ends of a string
TrimEnd	Trims specified characters from the end of a string
TrimStart	Trims specified characters from the start of a string

Fields

Empty	String	Represents the empty string. This field is read-only.

Properties

Chars	Char	Gets the character at a specified character position in this instance.
Length	Int	Gets the number of characters in this instance.

Operators

=	Determines whether two specified String objects have the same value.
<>	Determines whether two specified String objects have different values.

http://www.delphibasics.co.uk/NameSpace.asp?Name=System&Part=String

การจัด String Format

```
function Format ( Const Formatting : string; Const Data : array of const ) : string;
function Format ( Const Formatting : string; Const Data : array of const; FormatSettings : TFormatSettings ) : string;
```

```
var
  text : string;
begin
  // Just 1 data item
  ShowMessage(Format('%s', ['Hello']));

  // A mix of literal text and a data item
  ShowMessage(Format('String = %s', ['Hello']));
  ShowMessage('');
  // Examples of each of the data types
  ShowMessage(Format('Decimal = %d', [-123]));
  ShowMessage(Format('Exponent = %e', [12345.678]));
  ShowMessage(Format('Fixed = %f', [12345.678]));
  ShowMessage(Format('General = %g', [12345.678]));
  ShowMessage(Format('Number = %n', [12345.678]));
  ShowMessage(Format('Money = %m', [12345.678]));
  ShowMessage(Format('Pointer = %p', [addr(text)]));
  ShowMessage(Format('String = %s', ['Hello']));
  ShowMessage(Format('Unsigned decimal = %u', [123]));
  ShowMessage(Format('Hexadecimal = %x', [140]));
end;
```

Results

```
Hello
String = Hello
Decimal = -123
Exponent = 1.23456780000000E+004
Fixed = 12345.68
General = 12345.678
```

```
Number = 12,345,68
Money = ?12,345.68
Pointer = 0069FC90
String = Hello
Unsigned decimal = 123
Hexadecimal = 8C
```

The Formatting string can comprise a mix of ordinary characters (that are passed unchanged to the result string), and data formatting characters. This formatting is best explained by the example code.

In simple terms, each data formatting substring starts with a % and ends with a data type indicator :

```
d = Decimal (integer)
e = Scientific
f = Fixed
g = General
m = Money
```

```
n = Number (floating)
p = Pointer
s = String
u = Unsigned decimal
x = Hexadecimal
```

http://www.delphibasics.co.uk/RTL.asp?Name=Format

การแปลงชนิดข้อมูลเป็นการแปลงข้อมูลจากชชิดหนึ่งไปเป็นข้อมูลอีกชนิดหนึ่งซึ่งข้อมูลต้นทางจะต้องมีรูปแบบที่เหมาะสมก่อนแปลงเสมอ ถ้าข้อมูลต้นทางไม่สามารถแปลงเป็นชนิดข้อมูลปลายทางได้จะเกิด Error เช่น Function ใช้ในการแปลงชนิดข้อมูลถูกเก็บไว้ใน SysUtils unit

```
'1' แปลงเป็นเลข 1 ได้ เช่น x:=StrToInt('1');
'a' ไม่สามารถแปลงเป็นตัวเลขได้ x:=StrToInt('a'); // Error
```

http://www.delphibasics.co.uk/RTL.asp?Name=StrToInt

3. Pointer

```
var
  generalPtr : Pointer;  // A pointer to anything
  formPtr    : ^TForm;   // A pointer to a form object

begin
 // The current unit's form is addressable via the self
keyword
  generalPtr := Addr(self);

  // We can assign this pointer to the form pointer
  formPtr := generalPtr;

  // And set the form caption to show this
  formPtr.Caption := 'Test program';
end;
```

```
  The form is shown with caption:
  Test program
```

http://www.delphibasics.co.uk/RTL.asp?Name=Pointer

```
var
X, Y: Integer;  // X and Y are Integer variables
P: ^Integer;     // P points to an Integer
begin
X := 17;      // assign a value to X
P := @X;      // assign the address of X to P
Y := P^;      // dereference P; assign the result to Y
end;
```

http://docwiki.embarcadero.com/RADStudio/Rio/en/Pointers_and_Pointer_Types_(Delphi)

http://rvelthuis.de/articles/articles-pointers.html

```
type
 TDataPackage = record
  Header: array [0..1] of Byte;
  Id:Byte;
  Len:Byte;
  Func:Byte;
  Para: array [0..254] of Byte;
  ChkSum:Byte;
end;

procedure TForm1.Button5Click(Sender: TObject);
begin
 DataPackage.Header[0]:=$31;
 DataPackage.Header[1]:=$32;
```

```
Label6.Caption:=ArrayByteToString(@DataPackage.Head
er, Sizeof(DataPackage.Header));
end;

function TForm1.ArrayByteToString(MyByte: PByte;
Len:Integer): String;
var
 Ans:String;
 i:Integer;
```

```
begin
 Ans:='';
 for i := 0 to Len-1 do
  begin
   Ans:=Ans+Char(MyByte^);
   Inc(MyByte);
  end;
 Result:=Ans;
end;
```

```
type
  TMyRecord = Record
    name : String[20];
    age  : Integer;
  end;

var
  myRecord    : TMyRecord;
  myRecordPtr : ^TMyRecord;
 x:String;
```

```
begin
  myRecord.name := 'Fred Bloggs';
  myRecord.age  := 23;

  myRecordPtr := @myRecord;
  x:=myRecordptr^.name;     // OK
 ShowMessage(x);
 x:=myRecordptr.name;       // OK
 ShowMessage(x);// Displays 'Fred Bloggs'
end;
```

http://www.delphibasics.co.uk/Article.asp?Name=Pointers

4. Date-Time

Date and time calculations

DayOfTheMonth	Gives the day of month index for a TDateTime value
DaysBetween	Gives the whole number of days between 2 dates
DaysInAMonth	Gives the number of days in a month
DaysInAYear	Gives the number of days in a year
DecodeDate	Extracts the year, month, day values from a TDateTime var.
EncodeDate	Build a TDateTime value from year, month and day values
IncDay	Increments a TDateTime variable by + or - number of days
IsLeapYear	Returns true if a given calendar year

Displaying date and time values

```
var
  myDate : TDateTime;

begin
  // Set up our TDateTime variable with a full date and time :
  // 09/02/2000 at 05:06:07.008 (.008 milli-seconds)
  myDate := EncodeDateTime(2000, 2, 9, 5, 6, 7, 8);
// Date only - numeric values with no leading zeroes (except year)
ShowMessage(' d/m/y = '+ FormatDateTime('d/m/y', myDate));

  // Date only - numeric values with leading zeroes
ShowMessage('          dd/mm/yy = '+ FormatDateTime('dd/mm/yy', myDate));
  // Use short names for the day, month, and add freeform text ('of')
ShowMessage('  ddd d of mmm yyyy = '+ FormatDateTime('ddd d of mmm yyyy', myDate));

  // Use long names for the day and month
ShowMessage('dddd d of mmmm yyyy = '+ FormatDateTime('dddd d of mmmm yyyy', myDate));
```

```
  // Use the ShortDateFormat settings only
  ShowMessage('ddddd = '+FormatDateTime('ddddd', myDate));

  // Use the LongDateFormat settings only
  ShowMessage('dddddd = '+FormatDateTime('dddddd', myDate));
  ShowMessage('');
  // Time only - numeric values with no leading zeroes
  ShowMessage('h:n:s.z = '+FormatDateTime('h:n:s.z', myDate));
  // Time only - numeric values with leading zeroes
  ShowMessage('hh:nn:ss.zzz = '+FormatDateTime('hh:nn:ss.zzz', myDate));
  // Use the ShortTimeFormat settings only
  ShowMessage('t = '+FormatDateTime('t', myDate));
  // Use the LongTimeFormat settings only
  ShowMessage(' tt = '+FormatDateTime('tt', myDate));

  // Use the ShortDateFormat + LongTimeFormat settings
  ShowMessage('c = '+FormatDateTime('c', myDate));
end;
```

results

```
The ShowMessage routine shows the following outputs :
d/m/y = 9/2/00
dd/mm/yy = 09/02/00
ddd d of mmm yyyy = Wed 9 of Feb 2000
dddd d of mmmm yyyy = Wednesday 9 of February 2000
ddddd = 09/02/2000
```

```
dddddd = 09 February 2000
c = 09/02/2000 05:06:07
 h:n:s.z = 5:6:7.008
hh:nn:ss.zzz = 05:06:07.008
t = 05:06
tt = 05:06:07
c = 09/02/2000 05:06:07
```

http://www.delphibasics.co.uk/Article.asp?Name=Dates

Date-Time Function

Function	*Date*	*Gives the current date*
Variable	*DateSeparator*	*The character used to separate display date fields*
Function	*DateTimeToFileDate*	*Convert a TDateTime value to a File date/time format*
Function	*DateTimeToStr*	*Converts TDateTime date and time values to a string*
Procedure	*DateTimeToString*	*Rich formatting of a TDateTime variable into a string*
Function	*DateToStr*	*Converts a TDateTime date value to a string*
Function	*DayOfTheMonth*	*Gives day of month index for a TDateTime value (ISO 8601)*
Function	*DayOfTheWeek*	*Gives day of week index for a TDateTime value (ISO 8601)*
Function	*DayOfTheYear*	*Gives the day of the year for a TDateTime value (ISO 8601)*
Function	*DayOfWeek*	*Gives day of week index for a TDateTime value*
Function	*DaysBetween*	*Gives the whole number of days between 2 dates*
Function	*DaysInAMonth*	*Gives the number of days in a month*
Function	*DaysInAYear*	*Gives the number of days in a year*
Function	*DaySpan*	*Gives the fractional number of days between 2 dates*
Procedure	*DecodeDate*	*Extracts the year, month, day values from a TDateTime var.*
Procedure	*DecodeDateTime*	*Breaks a TDateTime variable into its date/time parts*
Procedure	*DecodeTime*	*Break a TDateTime value into individual time values*
Function	*EncodeDate*	*Build a TDateTime value from year, month and day values*
Function	*EncodeDateTime*	*Build a TDateTime value from day and time values*
Function	*EncodeTime*	*Build a TDateTime value from hour, min, sec and msec values*
Function	*EndOfADay*	*Generate a TDateTime value set to the very end of a day*
Function	*EndOfAMonth*	*Generate a TDateTime value set to the very end of a month*
Function	*FileAge*	*Get the last modified date/time of a file without opening it*
Function	*FileDateToDateTime*	*Converts a file date/time format to a TDateTime value*
Function	*FileSetDate*	*Set the last modified date and time of a file*
Function	*FormatDateTime*	*Rich formatting of a TDateTime variable into a string*
Function	*IncDay*	*Increments a TDateTime variable by + or - number of days*
Function	*IncMillisecond*	*Increments a TDateTime variable by + or - number of milliseconds*
Function	*IncMinute*	*Increments a TDateTime variable by + or - number of minutes*
Function	*IncMonth*	*Increments a TDateTime variable by a number of months*
Function	*IncSecond*	*Increments a TDateTime variable by + or - number of seconds*
Function	*IncYear*	*Increments a TDateTime variable by a number of years*
Function	*IsLeapYear*	*Returns true if a given calendar year is a leap year*
Variable	*LongDateFormat*	*Long version of the date to string format*
Variable	*LongDayNames*	*An array of days of the week names, starting 1 = Sunday*
Variable	*LongMonthNames*	*An array of days of the month names, starting 1 = January*
Variable	*LongTimeFormat*	*Long version of the time to string format*

Constant	MinsPerDay	Gives the number of minutes in a day
Constant	MonthDays	Gives the number of days in a month
Function	MonthOfTheYear	Gives the month of the year for a TDateTime value
Function	Now	Gives the current date and time
Function	RecodeDate	Change only the date part of a TDateTime variable
Function	RecodeTime	Change only the time part of a TDateTime variable
Procedure	ReplaceDate	Change only the date part of a TDateTime variable
Procedure	ReplaceTime	Change only the time part of a TDateTime variable
Constant	SecsPerDay	Gives the number of seconds in a day
Variable	ShortDateFormat	Compact version of the date to string format
Variable	ShortDayNames	An array of days of the week names, starting 1 = Sunday
Variable	ShortMonthNames	An array of days of the month names, starting 1 = Jan
Variable	ShortTimeFormat	Short version of the time to string format
Function	StrToDate	Converts a date string into a TDateTime value
Function	StrToDateTime	Converts a date+time string into a TDateTime value
Function	StrToTime	Converts a time string into a TDateTime value
Type	TDateTime	Data type holding a date and time value
Function	Time	Gives the current time
Variable	TimeAMString	Determines AM value in DateTimeToString procedure
Variable	TimePMString	Determines PM value in DateTimeToString procedure
Variable	TimeSeparator	The character used to separate display time fields
Function	TimeToStr	Converts a TDateTime time value to a string
Function	Tomorrow	Gives the date tomorrow
Variable	TwoDigitYearCenturyWindow	Sets the century threshold for 2 digit year string conversions
Function	Yesterday	Gives the date yesterday

http://www.delphibasics.co.uk/ByFunction.asp?Main=DatesAndTimes

Example

```
var
  day1, day2 : TDateTime;
  diff : Double;
begin
  day1 := StrToDate('12/06/2002');
  day2 := StrToDate('12/07/2002');
  ShowMessage('day1 = '+DateToStr(day1));
  ShowMessage('day2 = '+DateToStr(day2));
  diff := day2 - day1;
  ShowMessage('day2 - day1 = '+FloatToStr(diff)+' days');
end;
```

results

```
day1 = 12/06/2002
```

```
day2 = 12/07/2002
```

```
day2 - day1 = 30 days
```

http://www.delphibasics.co.uk/RTL.asp?Name=TDateTime

5. ใช้ Function สำเร็จรูปเกี่ยวกับข้อมูล

Strings and Chars

Function	AnsiCompareStr	Compare two strings for equality
Function	AnsiCompareText	Compare two strings for equality, ignoring case
Function	AnsiContainsStr	Returns true if a string contains a substring
Function	AnsiContainsText	Returns true if a string contains a substring, case insensitive
Function	AnsiEndsStr	Returns true if a string ends with a substring
Function	AnsiIndexStr	Compares a string with a list of strings - returns match index
Function	AnsiLeftStr	Extracts characters from the left of a string
Function	AnsiLowerCase	Change upper case characters in a string to lower case
Function	AnsiMatchStr	Returns true if a string exactly matches one of a list of strings
Function	AnsiMidStr	Returns a substring from the middle characters of a string
Function	AnsiPos	Find the position of one string in another
Function	AnsiReplaceStr	Replaces a part of one string with another
Function	AnsiReverseString	Reverses the sequence of letters in a string
Function	AnsiRightStr	Extracts characters from the right of a string
Function	AnsiStartsStr	Returns true if a string starts with a substring
Function	AnsiUpperCase	Change lower case characters in a string to upper case
Procedure	AppendStr	Concatenate one string onto the end of another
Function	CompareStr	Compare two strings to see which is greater than the other
Function	CompareText	Compare two strings for equality, ignoring case
Function	Concat	Concatenates one or more strings into one string
Function	Copy	Create a copy of part of a string or an array
Variable	CurrencyDecimals	Defines decimal digit count in the Format function
Variable	CurrencyFormat	Defines currency string placement in curr display functions
Variable	CurrencyString	The currency string used in currency display functions
Function	CurrToStr	Convert a currency value to a string
Function	CurrToStrF	Convert a currency value to a string with formatting
Function	DateTimeToStr	Converts TDateTime date and time values to a string
Procedure	DateTimeToString	Rich formatting of a TDateTime variable into a string
Function	DateToStr	Converts a TDateTime date value to a string
Variable	DecimalSeparator	The character used to display the decimal point
Procedure	Delete	Delete a section of characters from a string

Function	*DupeString*	*Creates a string containing copies of a substring*
Function	*ExtractFileDir*	*Extracts the dir part of a full file name*
Function	*ExtractFileDrive*	*Extracts the drive part of a full file name*
Function	*ExtractFileExt*	*Extracts the extension part of a full file name*
Function	*ExtractFileName*	*Extracts the name part of a full file name*
Function	*FloatToStr*	*Convert a floating point value to a string*
Function	*FloatToStrF*	*Convert a floating point value to a string with formatting*
Function	*Format*	*Rich formatting of numbers and text into a string*
Function	*FormatCurr*	*Rich formatting of a currency value into a string*
Function	*FormatDateTime*	*Rich formatting of a TDateTime variable into a string*
Function	*FormatFloat*	*Rich formatting of a floating point number into a string*
Function	*High*	*Returns the highest value of a type or variable*
Function	*InputBox*	*Display a dialog that asks for user text input, with default*
Function	*InputQuery*	*Display a dialog that asks for user text input*
Procedure	*Insert*	*Insert a string into another string*
Function	*IntToHex*	*Convert an Integer into a hexadecimal string*
Function	*IntToStr*	*Convert an integer into a string*
Function	*LastDelimiter*	*Find the last position of selected characters in a string*
Function	*Length*	*Return the number of elements in an array or string*
Variable	*LongDateFormat*	*Long version of the date to string format*
Variable	*LongDayNames*	*An array of days of the week names, starting 1 = Sunday*
Variable	*LongMonthNames*	*An array of days of the month names, starting 1 = January*
Variable	*LongTimeFormat*	*Long version of the time to string format*
Function	*LowerCase*	*Change upper case characters in a string to lower case*
Function	*MessageDlg*	*Displays a message, symbol, and selectable buttons*
Function	*MessageDlgPos*	*Displays a message plus buttons at a given screen position*
Procedure	*Move*	*Copy bytes of data from a source to a destination*
Variable	*NegCurrFormat*	*Defines negative amount formatting in currency displays*
Function	*Ord*	*Provides the Ordinal value of an integer, character or enum*
Function	*Pos*	*Find the position of one string in another*
Procedure	*ProcessPath*	*Split a drive/path/filename string into its constituent parts*
Procedure	*SetLength*	*Changes the size of a string, or the size(s) of an array*
Procedure	*SetString*	*Copies characters from a buffer into a string*
Variable	*ShortDateFormat*	*Compact version of the date to string format*
Variable	*ShortDayNames*	*An array of days of the week names, starting 1 = Sunday*
Variable	*ShortMonthNames*	*An array of days of the month names, starting 1 = Jan*
Variable	*ShortTimeFormat*	*Short version of the time to string format*

Procedure	*ShowMessage*	*Display a string in a simple dialog with an OK button*
Procedure	*ShowMessageFmt*	*Display formatted data in a simple dialog with an OK button*
Procedure	*ShowMessagePos*	*Display a string in a simple dialog at a given screen position*
Procedure	*Str*	*Converts an integer or floating point number to a string*
Function	*StringOfChar*	*Creates a string with one character repeated many times*
Function	*StringReplace*	*Replace one or more substrings found within a string*
Function	*StringToWideChar*	*Converts a normal string into a WideChar 0 terminated buffer*
Function	*StrScan*	*Searches for a specific character in a constant string*
Function	*StrToCurr*	*Convert a number string into a currency value*
Function	*StrToDate*	*Converts a date string into a TDateTime value*
Function	*StrToDateTime*	*Converts a date+time string into a TDateTime value*
Function	*StrToFloat*	*Convert a number string into a floating point value*
Function	*StrToInt*	*Convert an integer string into an Integer value*
Function	*StrToInt64*	*Convert an integer string into an Int64 value*
Function	*StrToInt64Def*	*Convert a string into an Int64 value with default*
Function	*StrToIntDef*	*Convert a string into an Integer value with default*
Function	*StrToTime*	*Converts a time string into a TDateTime value*
Function	*StuffString*	*Replaces a part of one string with another*
Type	*TFloatFormat*	*Formats for use in floating point number display functions*
Type	*TFormatSettings*	*A record for holding locale values for thread-safe functions*
Variable	*ThousandSeparator*	*The character used to display the thousands separator*
Function	*TimeToStr*	*Converts a TDateTime time value to a string*
Type	*TPrintDialog*	*Class that creates a printer selection and control dialog*
Type	*TReplaceFlags*	*Defines options for the StringReplace routine*
Function	*Trim*	*Removes leading and trailing blanks from a string*
Function	*TrimLeft*	*Removes leading blanks from a string*
Function	*TrimRight*	*Removes trailing blanks from a string*
Type	*TStringList*	*Holds a variable length list of strings*
Variable	*TwoDigitYearCenturyWindow*	*Sets the century threshold for 2 digit year string conversions*
Function	*UpCase*	*Convert a Char value to upper case*
Function	*UpperCase*	*Change lower case characters in a string to upper case*
Procedure	*Val*	*Converts number strings to integer and floating point values*
Function	*WideCharToString*	*Copies a null terminated WideChar string to a normal string*
Function	*WrapText*	*Add line feeds into a string to simulate word wrap*

http://www.delphibasics.co.uk/ByFunction.asp?Main=Strings

Numbers and Sets

Function	*Abs*	*Gives the absolute value of a number (-ve sign is removed)*
Function	*Addr*	*Gives the address of a variable, function or procedure*
Keyword	*And*	*Boolean and or bitwise and of two arguments*
Function	*ArcCos*	*The Arc Cosine of a number, returned in radians*
Function	*ArcSin*	*The Arc Sine of a number, returned in radians*
Function	*ArcTan*	*The Arc Tangent of a number, returned in radians*
Function	*Bounds*	*Create a TRect value from top left and size values*
Function	*CelsiusToFahrenheit*	*Convert a celsius temperature into fahrenheit*
Function	*Chr*	*Convert an integer into a character*
Function	*CompareValue*	*Compare numeric values with a tolerance*
Function	*Convert*	*Convert one measurement value to another*
Function	*Copy*	*Create a copy of part of a string or an array*
Function	*Cos*	*The Cosine of a number*
Variable	*CurrencyDecimals*	*Defines decimal digit count in the Format function*
Variable	*CurrencyFormat*	*Defines currency string placement in curr display functions*
Variable	*CurrencyString*	*The currency string used in currency display functions*
Function	*CurrToStr*	*Convert a currency value to a string*
Function	*CurrToStrF*	*Convert a currency value to a string with formatting*
Function	*DayOfTheMonth*	*Gives day of month index for a TDateTime value (ISO 8601)*
Function	*DayOfTheWeek*	*Gives day of week index for a TDateTime value (ISO 8601)*
Function	*DayOfTheYear*	*Gives the day of the year for a TDateTime value (ISO 8601)*
Function	*DayOfWeek*	*Gives day of week index for a TDateTime value*
Function	*DaysBetween*	*Gives the whole number of days between 2 dates*
Function	*DaysInAMonth*	*Gives the number of days in a month*
Function	*DaysInAYear*	*Gives the number of days in a year*
Function	*DaySpan*	*Gives the fractional number of days between 2 dates*
Procedure	*Dec*	*Decrement an ordinal variable*
Variable	*DecimalSeparator*	*The character used to display the decimal point*
Procedure	*DecodeDate*	*Extracts the year, month, day values from a TDateTime var.*
Procedure	*DecodeDateTime*	*Breaks a TDateTime variable into its date/time parts*
Procedure	*DecodeTime*	*Break a TDateTime value into individual time values*
Function	*DegToRad*	*Convert a degrees value to radians*
Keyword	*Div*	*Performs integer division, discarding the remainder*
Function	*EncodeDate*	*Build a TDateTime value from year, month and day values*
Function	*EncodeDateTime*	*Build a TDateTime value from day and time values*
Function	*EncodeTime*	*Build a TDateTime value from hour, min, sec and msec values*

Procedure	*Exclude*	*Exclude a value in a set variable*
Function	*Exp*	*Gives the exponent of a number*
Function	*FahrenheitToCelsius*	*Convert a fahrenheit temperature into celsius*
Procedure	*FillChar*	*Fills out a section of storage with a fill character or byte value*
Function	*FloatToStr*	*Convert a floating point value to a string*
Function	*FloatToStrF*	*Convert a floating point value to a string with formatting*
Function	*Format*	*Rich formatting of numbers and text into a string*
Function	*FormatCurr*	*Rich formatting of a currency value into a string*
Function	*FormatFloat*	*Rich formatting of a floating point number into a string*
Function	*Frac*	*The fractional part of a floating point number*
Function	*Hi*	*Returns the hi-order byte of a (2 byte) Integer*
Function	*High*	*Returns the highest value of a type or variable*
Keyword	*In*	*Used to test if a value is a member of a set*
Procedure	*Inc*	*Increment an ordinal variable*
Procedure	*Include*	*Include a value in a set variable*
Constant	*Infinity*	*Floating point value of infinite size*
Function	*Int*	*The integer part of a floating point number as a float*
Function	*IntToHex*	*Convert an Integer into a hexadecimal string*
Function	*IntToStr*	*Convert an integer into a string*
Function	*IsInfinite*	*Checks whether a floating point number is infinite*
Function	*IsNaN*	*Checks to see if a floating point number holds a real number*
Function	*Length*	*Return the number of elements in an array or string*
Function	*Ln*	*Gives the natural logarithm of a number*
Function	*Lo*	*Returns the low-order byte of a (2 byte) Integer*
Function	*Log10*	*Gives the log to base 10 of a number*
Function	*Low*	*Returns the lowest value of a type or variable*
Function	*Max*	*Gives the maximum of two integer values*
Constant	*MaxInt*	*The maximum value an Integer can have*
Constant	*MaxLongInt*	*The maximum value an LongInt can have*
Function	*Mean*	*Gives the average for a set of numbers*
Function	*Min*	*Gives the minimum of two integer values*
Constant	*MinsPerDay*	*Gives the number of minutes in a day*
Keyword	*Mod*	*Performs integer division, returning the remainder*
Constant	*MonthDays*	*Gives the number of days in a month*
Function	*MonthOfTheYear*	*Gives the month of the year for a TDateTime value*
Procedure	*Move*	*Copy bytes of data from a source to a destination*
Constant	*NaN*	*Not a real number*

Variable	*NegCurrFormat*	*Defines negative amount formatting in currency displays*
Function	*Odd*	*Tests whether an integer has an odd value*
Keyword	*Or*	*Boolean or or bitwise or of two arguments*
Function	*Ord*	*Provides the Ordinal value of an integer, character or enum*
Function	*Pi*	*The mathematical constant*
Function	*Point*	*Generates a TPoint value from X and Y values*
Function	*PointsEqual*	*Compares two TPoint values for equality*
Function	*Pred*	*Decrement an ordinal variable*
Function	*PtInRect*	*Tests to see if a point lies within a rectangle*
Function	*RadToDeg*	*Converts a radian value to degrees*
Function	*Random*	*Generate a random floating point or integer number*
Procedure	*Randomize*	*Reposition the Random number generator next value*
Function	*RandomRange*	*Generate a random integer number within a supplied range*
Variable	*RandSeed*	*Reposition the Random number generator next value*
Function	*RecodeDate*	*Change only the date part of a TDateTime variable*
Function	*RecodeTime*	*Change only the time part of a TDateTime variable*
Function	*Rect*	*Create a TRect value from 2 points or 4 coordinates*
Function	*Round*	*Rounds a floating point number to an integer*
Constant	*SecsPerDay*	*Gives the number of seconds in a day*
Procedure	*SetLength*	*Changes the size of a string, or the size(s) of an array*
Keyword	*Shl*	*Shift an integer value left by a number of bits*
Keyword	*Shr*	*Shift an integer value right by a number of bits*
Function	*Sin*	*The Sine of a number*
Function	*SizeOf*	*Gives the storage byte size of a type or variable*
Function	*Slice*	*Creates a slice of an array as an Open Array parameter*
Function	*Sqr*	*Gives the square of a number*
Function	*Sqrt*	*Gives the square root of a number*
Procedure	*Str*	*Converts an integer or floating point number to a string*
Function	*StrToCurr*	*Convert a number string into a currency value*
Function	*StrToFloat*	*Convert a number string into a floating point value*
Function	*StrToInt*	*Convert an integer string into an Integer value*
Function	*StrToInt64*	*Convert an integer string into an Int64 value*
Function	*StrToInt64Def*	*Convert a string into an Int64 value with default*
Function	*StrToIntDef*	*Convert a string into an Integer value with default*
Function	*Succ*	*Increment an ordinal variable*
Function	*Sum*	*Return the sum of an array of floating point values*
Function	*Tan*	*The Tangent of a number*

Type	*TBits*	*An object that can hold an infinite number of Boolean values*
Variable	*TConvFamily*	*Defines a family of measurement types as used by Convert*
Type	*TConvType*	*Defines a measurement type as used by Convert*
Type	*TFloatFormat*	*Formats for use in floating point number display functions*
Variable	*ThousandSeparator*	*The character used to display the thousands separator*
Type	*TPoint*	*Holds X and Y integer values*
Type	*TRect*	*Holds rectangle coordinate values*
Function	*Trunc*	*The integer part of a floating point number*
Procedure	*Val*	*Converts number strings to integer and floating point values*
Keyword	*Xor*	*Boolean Xor or bitwise Xor of two arguments*

http://www.delphibasics.co.uk/ByFunction.asp?Main=Numbers

Example

Code\week4\Ex_DataType_1

Code\week4\ Ex_Pointer_1

Exercise

- *อธิบายข้อแตกต่างระหว่างคำสั่ง for, while, repeat*
- *จงสร้างตัวแปรที่ใช้เก็บข้อมูลของเครื่องจักร 5 เครื่องโดยแต่ละเครื่องมีข้อมูลดังนี้ ชื่อ(String), เลขรหัส(Char), กระแสไฟฟ้า(Integer), แรงดันไฟฟ้า(Single) พร้อมแสดงวิธีการเก็บค่าและเรียกมาใช้งาน*

Assignment

- *เขียนโปรแกรมรับค่าข้อมูลจากเซ็นเซอร์เป็นเลขฐาน 16 จำนวน 5 Bytes ให้ Byte แรกคือเลขรหัส ซึ่งต้องแปลงให้เป็น Char ส่วน Byte ที่ 2-3 เก็บข้อมูลกระแสไฟฟ้า Byte ที่ 4-5 เก็บข้อมูลแรงดันไฟฟ้าโดยให้ป้อนข้อมูลค่าจากเซ็นเซอร์ที่ TEdit และนำค่าไปเก็บในตัวแปรที่สร้างจาก Exercise ที่2*

Answer Sheet

Code\week5\ Ans_for_1

Code\week5\ Ans_Record_1

Code\week5\ Ans_ByteDecode_1