# Records
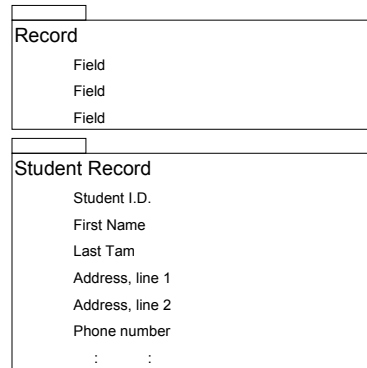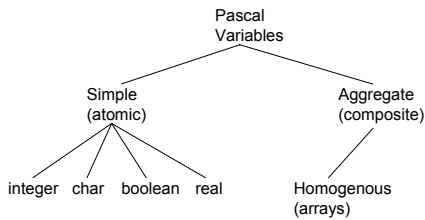
You will learn in this section of notes how to create a new, composite type, that can be composed of different types of elements.

---

## What Is A Record?

Record
- Field
- Field
- Field

Student Record
- Student I.D.
- First Name
- Last Tam
- Address, line 1
- Address, line 2
- Phone number
  - :       :

---

## Types Of Variables: What You Know

Pascal Variables
- Simple (atomic)
  - integer   char   boolean   real
- Aggregate (composite)
  - Homogenous (arrays)
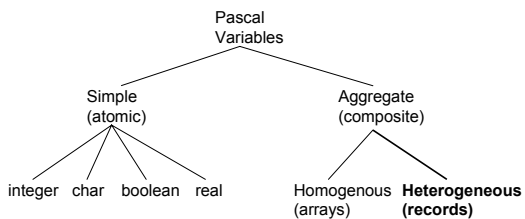
---

## Declaring Records

Format:
```
type
   Name of record = record
       name of field (1) : type of field (1);
       name of field (2) : type of field (2);
       name of field (3) : type of field (3);
          :   :   :        :   :   :
       name of field (n) : type of field (n);
   end; (* Record declaration *)
```

---

## Types Of Variables: What You Will Learn About

Pascal Variables
- Simple (atomic)
  - integer   char   boolean   real
- Aggregate (composite)
  - Homogenous (arrays)   **Heterogeneous (records)**

---
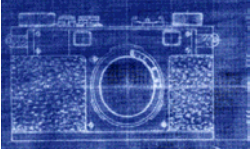
## Declaring Records (2)

Example:
```
const
   MAX_LENGTH = 20;
type
   StudentRecord = record
      studentIdentification : integer;
      firstName             : array [1..MAX_LENGTH] of char;
      lastName              : array [1..MAX_LENGTH] of char;
      phoneNumber           : integer;
   end;
```

## A Record Definition Is Like A Blueprint

•It indicates the format for what an example of the record should look like (what attribute fields will exist)
•No record is actually created by this definition
•No memory is allocated.

---

## Declaring Variables That Are Records

Format:
   *name of variable* : *name of declared record*;

Example:
   var jamesTam    : StudentRecord;
   var bartSimpson : StudentRecord;

| jamesTam |
| --- |

| bartSimpson |
| --- |

---

## Declaring Variables That Are Records

Format:
   *name of variable* : *name of record*;

Example:
   var jamesTam     : StudentRecord;
   var bartSimpson  : StudentRecord;

---

## Using Record Variables

Example: Declaring the record and instances of the record
```
const
  MAX_LENGTH = 20;
type
  StudentRecord = record
      studentIdentification : integer;
      firstName             : array [1..MAX_LENGTH] of char;
      lastName              : array [1..MAX_LENGTH] of char;
      phoneNumber           : integer;
  end;
begin
   var jamesTam    : StudentRecord;
   var bartSimpson : StudentRecord;
   :       :                :
end.
```

---

## Declaring An Instance Of A Record Actually Creates A Record

•Something has now been actually created.

---

## Using Record Variables (2)

Assignment (field-by-field basis):
e.g.,
```
     bartSimpson.studentIdentification := 123456;
     bartSimpson.firstName := 'bart';
     bartSimpson.lastName := 'simpson';
     bartSimpson.phoneNumber :=1234567;
```

Assignment (entire record – if the records are declared to be the same type)
e.g.,
```
     jamesTam := bartSimpson;
```

## Using Record Variables (3)

•Input and output via read/readln and write/writeln

•Must be done on a field by field basis (if the field is a type that can be "understood"[1] by read/readln or write/writeln

e.g.,

```
write('Enter ID for student : ');
readln(jamesTam.studentIdentification);

writeln('First name: ', jamesTam.firstName);
```

## Declaring Arrays Of Records

```
const
  MAX_LENGTH = 20;
  MAX_STUDENTS = 30000;
type
  StudentRecord = record
      studentIdentification : integer;
      firstName             : array [1..MAX_LENGTH] of char;
      lastName              : array [1..MAX_LENGTH] of char;
      phoneNumber           : integer;
  end;

  StudentRecordList = array [1..MAX_STUDENTS] of StudentRecord;

         :              :              :
  var universityOfCalgaryStudentRecords : StudentRecordList;
```

## A Shortcut For Referencing All The Fields Of A Record: With-Do

•Allows you to refer to the fields of a record without having to constantly refer to the name of the record variable.

Format:

```
with name of record variable do
    body
```

Example:

```
with bartSimpson do
begin
    writeln('Student record:');
    writeln('First name: ':11, firstName);
    writeln('Last name: ':11, lastName);
    writeln('ID number: ':11, studentIdentification);
    writeln('Phone: ':11, phoneNumber);
end; (* With do for Bart Simpson *)
```

## Declaring Arrays Of Records

```
const                                          1. Declaring a
  MAX_LENGTH = 20;                                 new Record
  MAX_STUDENTS = 30000;
type
  StudentRecord = record
      studentIdentification : integer;
      firstName             : array [1..MAX_LENGTH] of char;  2. Declaring a
      lastName              : array [1..MAX_LENGTH] of char;     type for the
      phoneNumber           : integer;                          array of
  end;                                                          records

  StudentRecordList = array [1..MAX_STUDENTS] of StudentRecord;

         :              :              :         3. Declaring a
  var universityOfCalgaryStudentRecords : StudentRecordList;   new
                                                  instance of
                                                  the type
```

## Declaring Arrays Of Records

Method:

1) Declare the record
2) Declare a type for the array of records
3) Declare the array of records

*As with arrays of simple types, the second step is essential in Pascal for passing the array as a parameter into functions and procedures!*

## Passing Records And Arrays Of Records As Parameters

• Looks the same as passing in other types of variables
• Can be passed in as value or variable parameters

Examples (function or procedure call):
```
displayStudent (jamesTam);
initializeStudentRecords (universityOfCalgaryStudentRecords);
```

Examples (function or procedure definition)
```
procedure displayStudent (jamesTam : StudentRecord);
begin
end;  (* Procedure displayStudent *)

procedure initializeStudentRecords (var
    universityOfCalgaryStudentRecords : StudentRecordList);
begin
end; (* Procedure initializeStudentRecords *)
```

## Putting This All Together

You can find a full version of this program in Unix under:
/home/231/examples/records/person.p

```pascal
program person (input, output);

const
  NAME_LENGTH = 16;
  NO_PEOPLE  = 4;

type
  Person = Record
           name   : array [1..NAME_LENGTH] of char;
           age    : integer;
           height : real;
           weight : real;
         end; (* Declaration of Person *)
```

James Tam

## Putting This All Together (2)

```pascal
People = array [1..NO_PEOPLE] of Person;

procedure manualInitialization (var calgaryPeople : People );
var
  i : integer;
```

James Tam

## Putting This All Together (3)

```pascal
begin (* Start of manualInitialization *)
  for i := 1 to NO_PEOPLE do
  begin
    with calgaryPeople[i] do
    begin
      write('Enter name of person: ');
      readln(name);
      write('Enter age of person in whole years: ');
      readln(age);
      write('Enter the height of the person in inches: ');
      readln(height);
      write('Enter the weight of the person in pounds: ');
      readln(weight);
      writeln;
    end; (* With-do *)
  end; (* Initialization for-loop *)
end; (* End of manualInitialization *)
```

James Tam

## Putting It All Together (4)

```pascal
procedure fileInitialization (var peopleValues  : text;
                              var calgaryPeople : People);
var
  i : integer;
```

James Tam

## Putting It All Together (5)

```pascal
begin   (* Start of fileInitialization *)
  reset(peopleValues, 'peopleValues');
  writeln('Reading initial values from file "peopleValues"');
  for i := 1 to NO_PEOPLE do
  begin
    with calgaryPeople[i] do
    begin
      readln(peopleValues, name);
      readln(peopleValues, age);
      readln(peopleValues, height);
      readln(peopleValues, weight);
      readln(peopleValues);
    end; (* With-do *)
  end; (* Initialization for-loop *)
  close(peopleValues);
end; (* End of fileInitialization *)
```

James Tam

## Putting It All Together (6)

```pascal
procedure display (calgaryPeople : People);
var
  i : integer;
begin (* Start of display *)
  writeln;
  for i := 1 to NO_PEOPLE do
  begin
    with calgaryPeople[i] do
    begin
      writeln;
      writeln('Name: ', name);
      writeln('Age: ', age);
      writeln('Height: ', height:0:2);
      writeln('Weight: ', weight:0:2);
    end; (* With-do *)
  end; (* Display for-loop *)
  writeln;
end; (* End of display *)
```

James Tam

## Putting It All Together (7)

```
begin   (* Main program *)
   var peopleValues        : text;
   var calgaryPeople        : People;
   var initializationMethod : integer;

   writeln;
   writeln('Select method to set starting values for the people');
   writeln('Enter "1" to read the values in from a file');
   writeln('Enter "2" to manually enter in the values yourself');
   write('Enter your choice: ');
   readln(initializationMethod);
   writeln;
```

## You Should Now Know

- How to declare a record
- How to declare instances of records
- The difference between accessing an entire record and individual fields of a record and how each approach is done in Pascal
- How to work with arrays of records
  - How to declare an array of records
  - How to access individual array elements
  - Passing arrays of records as parameters
- How to use the with-do construct

## Putting It All Together (8)

```
case (initializationMethod) of
 1 :
 begin
     fileInitialization(peopleValues, calgaryPeople);
     display(calgaryPeople);
 end;

 2 :
 begin
     manualInitialization(calgaryPeople);
     display(calgaryPeople);
 end;

 else
 begin
    writeln('Your choice was not one of the available options.');
    writeln('Restart program and select again.');
 end; (* else case *)
```

## Putting It All Together (9)

```
  end; (* case *)
end.  (* program *)
```