Records

You will learn in this section of notes what is a record and how to use them in Pascal

Declaring Types

Why bother?

•Creating your own type of variable

•Making a synonym for an existing type

Syntax:

```
    Type
      Name(1) = Type for name (1);
      Name(2) = Type for name (2);
          :          :       :     :
      Name(n) = Type for name (n);
```

Declaring Types (2)

Example:

```
Type
    world = array [1..20,1..20] of char;
Var
     map        : world;
     biosphere: world;
```

Declaring Types (3)

Can be used to provide alternative names for existing types

Example:

```
type
    FloatingPoint = real;
var
    gpa      : FloatingPoint;
    income: real;
```

Declaring Types (3)

Can be used to provide alternative names for existing types

Example:

```
type
    FloatingPoint = real;
var
    gpa     : FloatingPoint;
    income: real;
```

Original type still usable

Declaring Types (4)

Example:

```
Type
    world = array [1..20,1..20] of char;
Var
    map         : world;
    biosphere: world;
```

Declaring the type - defining what the type consists of

Declaring variables of the new type

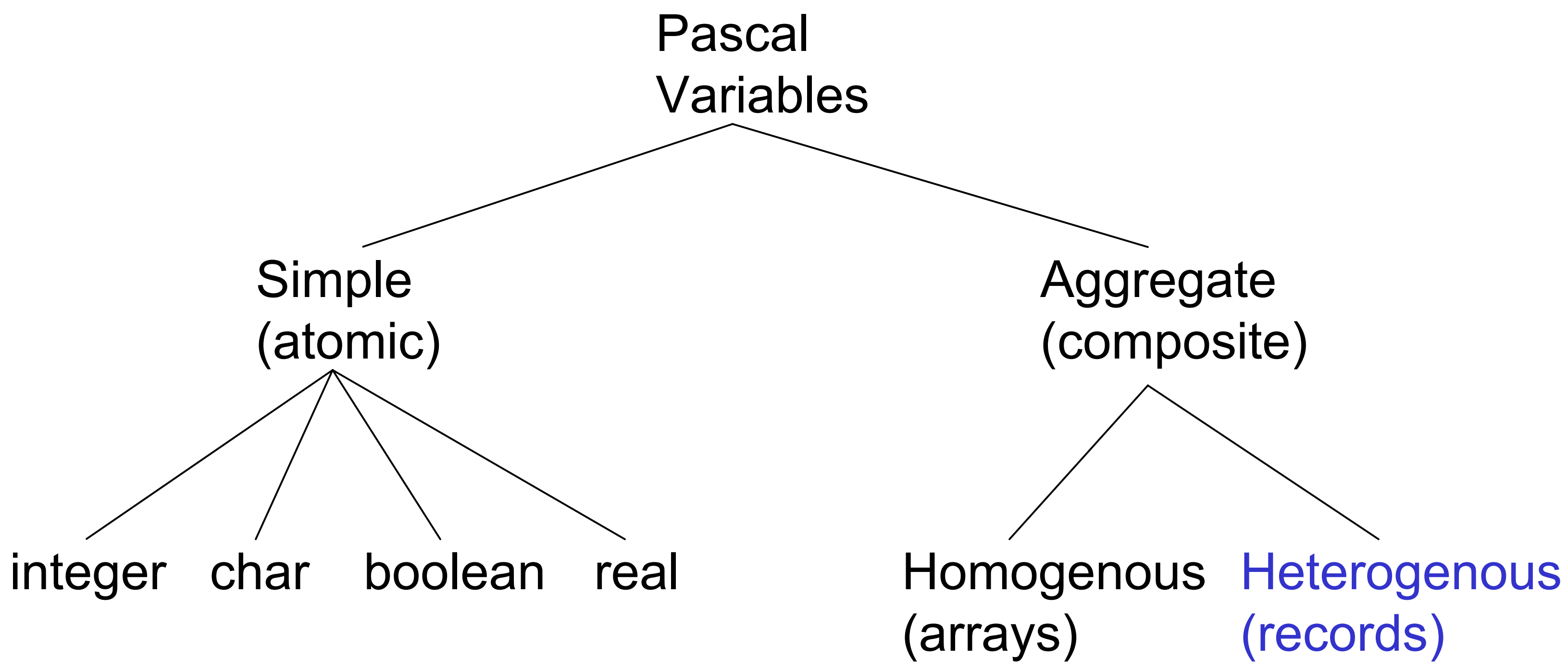
Where Type Declarations Fit Into Pascal Programs

```
program name;

(* Declarations *)
const  (* Declaration of constants)
type (* Declaration of new types *)
var    (* Declaration of variables *)
        (* Declarations of functions & procedures – defining
            what they do *)

begin
end.
```

Types Of Variables



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

: :

Declaring Records

Syntax:

```
Name = 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 *)
```

Declaring Records (2)

Example:

```
StudentRecord = record
    studentIdentification : integer;
    firstName             : array [1..20] of char;
    lastName              : array [1..20] of char;
    initial               : char;
    addressLineOne        : array [1..20] of char;
    addressLineTwo        : array [1..20] of char;
    phoneNumber           : integer;
end;
```

Declaring Variables That Are Records

Syntax:

name of variable : *name of declared record*;

Example:

```
jamesTam    : studentRecord;
bartSimpson : studentRecord;
```

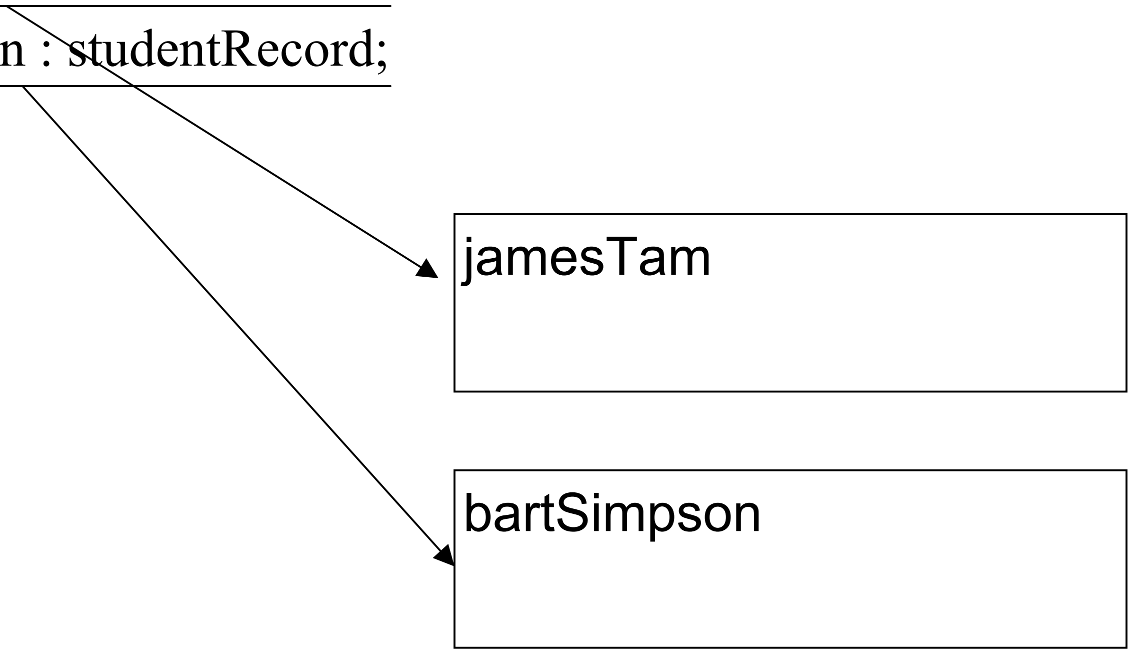
Declaring Variables That Are Records

Syntax:

name of variable : *name of declared record*;

Example:

```
jamesTam    : studentRecord;
bartSimpson : studentRecord;
```



Declaring Arrays Of Records

Method:

1) Declare the record

2) Declare a type for the array of records

The second step is essential in Pascal for passing the array as a parameter into functions and procedures!

Declaring Arrays Of Records

```
type
   StudentRecord = record
          studentIdentification : integer;
          firstName               : array [1..20] of char;
          lastName                : array [1..20] of char;
          initial                 : char;
          addressLineOne          : array [1..20] of char;
          addressLineTwo          : array [1..20] of char;
          phoneNumber             : integer;
    end;

    StudentRecordList = array [1..30000] of StudentRecord;
var
    universityOfCalgaryStudentRecords : StudentRecordList;
```

Declaring Arrays Of Records

```
type
 StudentRecord = record
        studentIdentification : integer;
        firstName              : array [1..20] of char;
        lastName               : array [1..20] of char;
        initial                : char;
        addressLineOne         : array [1..20] of char;
        addressLineTwo         : array [1..20] of char;
        phoneNumber            : integer;
  end;

  StudentRecordList = array [1..30000] of StudentRecord;
var
  universityOfCalgaryStudentRecords : StudentRecordList;
```

Declaring Record

Declaring a new type

Declaring a new instance of type "StudentRecordList"

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 calls):

```
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 *)
```

Returning Composite Types From Functions

- You cannot return composite types of variables (arrays and records) from functions.
- To have changes to these types of variables be retained after the function or procedure has ended they must be passed as variable parameters (example shown on previous slide)

Using Record Variables

Indicate which variable you wish to use (by stating the name of the variable)

e.g.,

```
type
   Person =  Record
                  name   : array [1..8] of char;
                  age      : integer;
                  height  : real;
                  weight : real;
               end; (* Declaration of Person *)
begin
     jo, jack : Person;
```

Using Record Variables (2)

(If applicable) indicate which field of the record that you wish to use.

e.g.,

```
jo.name := 'joanne';
jo.age := 20;
jo.height := 68.5;
jo.weight := 110;
```

Assignment

Can be done on a field by field basis:

e.g.,

```
jack.age = jo.age;
```

Can be done for the whole record (if the records are the same type)

e.g.,

```
jack := jo;
```

Using Record Variables (3)

Input and output via read/readln and write/writeln

Must be done on a field by field basis

e.g.,

```
write('Enter age for Jack : ');
 readln(jack.age);

writeln('Jack is ', jack.age, ' years old);
```

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.

Syntax:

```
with name do
    body
```

Example:

```
with jack do
begin
    writeln('Jack's stats');
    writeln('Age: ', age);
    writeln('Height :', height);
    writeln('Weight :', weight);
end; (* With do for jack *)
```

Putting This All Together

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

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

const
   NAMELENGTH = 16;
   NOPEOPLE   =  4;

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

  People = array [1..NOPEOPLE] of Person;
var
   peopleValues :  text;
```

Putting This All Together (2)

```
procedure manuallyInitializeCalgaryPeople (var calgaryPeople : People );
var
  i : integer;
begin
  for i := 1 to NOPEOPLE 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; (* Procedure manuallyInitializeCalgaryPeople *)
```

Putting This All Together (3)

```
procedure defaultInitializeCalgaryPeople (var calgaryPeople :  People);
var
  i          : integer;
begin
  reset(peopleValues);
  writeln('Reading initial values from file "peopleValues"');
  for i := 1 to NOPEOPLE 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; (* Procedure defaultInitializeCalgaryPeople *)
```

Putting It All Together (4)

```
procedure displayCalgaryPeople (calgaryPeople : People);
var
  i : integer;
begin
  writeln;
  for i := 1 to NOPEOPLE 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;
```

Putting It All Together (5)

```
begin
   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;
   case (initializationMethod) of
     1 :
     begin
          defaultInitializeCalgaryPeople(calgaryPeople);
          displayCalgaryPeople(calgaryPeople);
     end;

     2 :
     begin
         manuallyInitializeCalgaryPeople(calgaryPeople);
         displayCalgaryPeople(calgaryPeople);
     end;
```

Putting It All Together (6)

```
begin
   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;
   case (initializationMethod) of
     1 :
     begin
           defaultInitializeCalgaryPeople(calgaryPeople);
           displayCalgaryPeople(calgaryPeople);
     end;

     2 :
     begin
          manuallyInitializeCalgaryPeople(calgaryPeople);
          displayCalgaryPeople(calgaryPeople);
     end;
```

Putting It All Together (6)

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

Summary

How and why to declare new types in Pascal

What is a record

How do you declare a record

How do you declare variables that are records

Using records (accessing and assigning values)

Arrays of records

Records and arrays as parameters