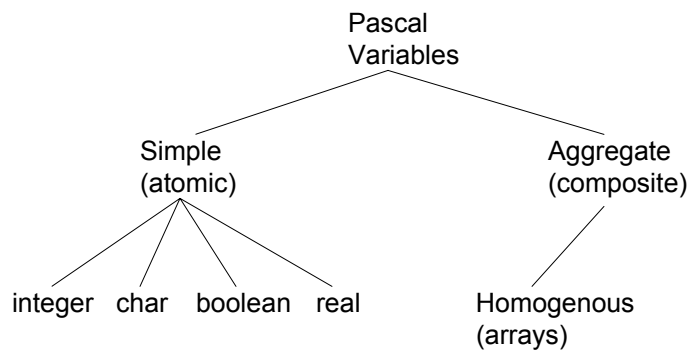


Records

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

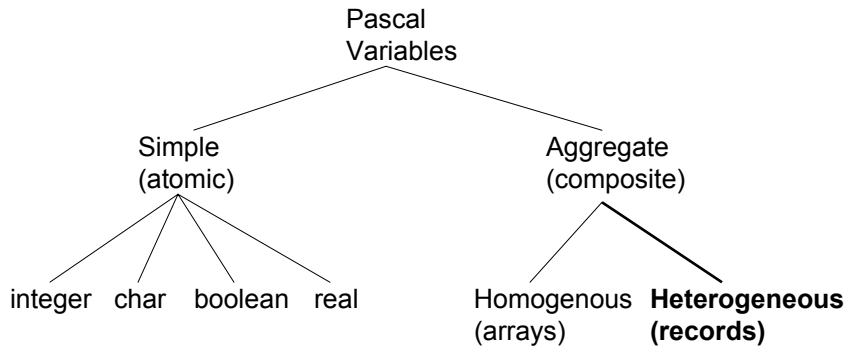
James Tam

Types Of Variables: What You Know



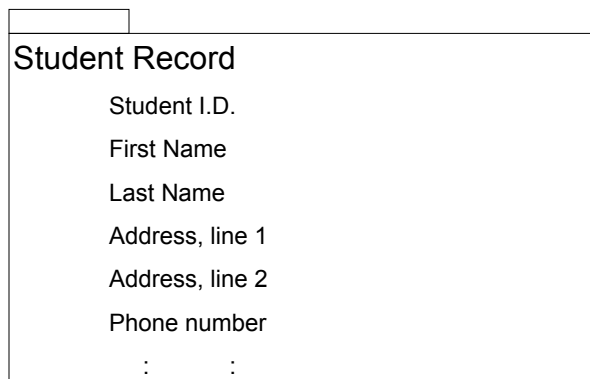
James Tam

Types Of Variables: What You Will Learn About



James Tam

What Is A Record?



James Tam

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 *)
```

James Tam

Declaring Records (2)

Example:

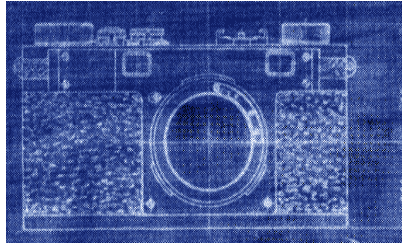
```
const
    NAME_LENGTH = 16;

type
    Person = record
        name : string [NAME_LENGTH];
        age : integer;
        height : real;
        weight : real;
    end; (* Declaration of Person *)
```

James Tam

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.



James Tam

Declaring Variables That Are Records

Format:

name of variable : name of record;

Example:

```
var jamesTam : Person;
```

```
var bartSimpson : Person;
```

James Tam

Declaring An Instance Of A Record Actually Creates A Record

- Something is now actually created.



James Tam

Declaring Variables That Are Records

Format:

name of variable : name of declared record;

Example:

var jamesTam : Person;

var bartSimpson : Person;

jamesTam

bartSimpson

James Tam

Using Record Variables

Example: Declaring the record and instances of the record

```
const
  NAME_LENGTH = 16;

type
  Person = record
    name  : string [NAME_LENGTH];
    age   : integer;
    height : real;
    weight : real;
  end; (* Declaration of Person *)

begin
  var jamesTam   : Person;
  var bartSimpson : Person;
  :             :
end.
```

James Tam

Using Record Variables (2)

Assignment (field-by-field basis):

e.g.,

```
bartSimpson.name := 'Bart';
bartSimpson.age  := 10;
bartSimpson.height := 48;
bartSimpson.weight := 80;
```

Assignment (entire record – if the records are declared to be the same type)

e.g.,

```
jamesTam := bartSimpson;
```

James Tam

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 name for student : ');  
readln(jamesTam.name);  
  
writeln('First name: ', jamesTam.name);
```

¹ This includes the built in simple types as well as character arrays

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('Personal information:');  
  writeln('Name: ':8, name);  
  writeln('Age: ':8, age);  
  writeln('Height: ':8, height);  
  writeln('Weight: ':8, weight);  
end; (* With do for Bart Simpson *)
```

Passing Records As Parameters

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

Example (procedure call):

```
displayPerson (jamesTam);
```

Example (procedure definition)

```
procedure displayPerson (jamesTam : Person);  
begin  
    (* Body of the procedure *)  
end; (* Procedure displayStudent *)
```

James Tam

Putting This All Together

You can find a full version of this program in Unix under:
/home/231/examples/records/person2.p

```
program person (input, output);  
  
const  
    NAME_LENGTH = 16;  
  
type  
    Person = Record  
        name : string [NAME_LENGTH];  
        age : integer;  
        height : real;  
        weight : real;  
    end; (* Declaration of Person *)
```

James Tam

Putting This All Together (2)

```
procedure initialize (var bart : Person;
                    var james : Person);
begin
  writeln;
  writeln('Setting the starting values');
  with bart do
  begin
    write('Name: ');
    readln(name);
    write('Age: ');
    readln(age);
    write('Height: ');
    readln(height);
    write('Weight: ');
    readln(weight);
  end;
  james := bart;
end;
```

James Tam

Putting This All Together (3)

```
procedure display (bart : Person;
                 james : Person);
begin
  writeln;
  writeln('BART');
  writeln('Name: ', bart.name);
  writeln('Age: ', bart.age);
  writeln('Height: ', bart.height:0:2);
  writeln('Weight: ', bart.weight:0:2);
  writeln;
  writeln('JAMES');
  with james do
  begin
    writeln('Name: ', name);
    writeln('Age: ', age);
    writeln('Height: ', height:0:2);
    writeln('Weight: ', weight:0:2);
  end;
end; (* display *)
```

James Tam

Putting This All Together (4)

```
begin
  var bart : Person;
  var james : Person;

  initialize(bart,james);
  display(bart,james);
end.
```

James Tam

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!

James Tam

Declaring Arrays Of Records

```
const
  NAME_LENGTH = 16;
  MAX_PEOPLE = 10;

type
  Person = Record
    name : string [NAME_LENGTH];
    age : integer;
    height : real;
    weight : real;
  end; (* Declaration of Person *)

  People = array [1..MAX_PEOPLE] of Person;
           :           :           :
var calgaryPeople : People;
```

James Tam

Declaring Arrays Of Records

```
const
  NAME_LENGTH = 16;
  MAX_PEOPLE = 10;

type
  Person = Record
    name : string [NAME_LENGTH];
    age : integer;
    height : real;
    weight : real;
  end; (* Declaration of Person *)

  People = array [1..MAX_PEOPLE] of Person;
           :           :           :
var calgaryPeople : People;
```

1. Declaring a new Record

2. Declaring a type for the array of records

3. Declaring a new instance of the type

James Tam

Passing Arrays Of Records As Parameters

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

Example (procedure call):

```
display (calgaryPeople,noPeople);
```

Example (procedure definition)

```
procedure display (calgaryPeople : People;
                  noPeople      : integer);
begin
    (* Body of the procedure *)
end; (* Procedure display *)
```

James Tam

Putting This All Together

You can find a full version of this program in Unix under:

```
/home/231/examples/records/person2.p
```

```
program person2 (input, output);
const
    NAME_LENGTH = 16;
    MAX_PEOPLE  = 10;
    FILE_NAME_LENGTH = 256;
type
    Person = Record
        name   : string [NAME_LENGTH];
        age    : integer;
        height : real;
        weight : real;
    end; (* Declaration of Person *)
    People = array [1..MAX_PEOPLE] of Person;
```

James Tam

Putting This All Together (2)

```
procedure displayMenu;  
begin  
  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: ');  
end;
```

James Tam

Putting This All Together (3)

```
procedure manualInitialization (var calgaryPeople : People;  
                               var noPeople      : integer);  
begin  
  for noPeople := 1 to MAX_PEOPLE do  
  begin  
    with calgaryPeople[noPeople] do
```

James Tam

Putting This All Together (4)

```
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; (* manualInitialization *)
```

James Tam

Putting This All Together (5)

```
procedure fileInitialization (var calgaryPeople : People;
                             var noPeople      : integer);
var
  peopleValues : text;
  i            : integer;
  fileName     : string[FILE_NAME_LENGTH];
begin
  write('Enter name of input file: ');
  readln(filename);
  reset(peopleValues,filename);
  writeln('Reading initial values from file ', filename);
  if EOF (peopleValues) then
  begin
    noPeople := 0;
    writeln('File ', filename, ' is empty, nothing to read.');
```

James Tam

Putting This All Together (6)

```
else
begin
  noPeople := 0;
  while NOT EOF (peopleValues) AND (noPeople < MAX_PEOPLE) do
  begin
    noPeople := noPeople + 1;
    with calgaryPeople[noPeople] do
    begin
      readln(peopleValues,name);
      readln(peopleValues,age);
      readln(peopleValues,height);
      readln(peopleValues,weight);
      readln(peopleValues);
    end; (* With-do *)
  end; (* readLoop *)
end; (* else *)
close(peopleValues);
end; (* fileInitialization *)
```

James Tam

Putting This All Together (7)

```
procedure display (calgaryPeople : People;
                  noPeople      : integer);
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; (* display *)
```

James Tam

Putting This All Together (8)

```
procedure handleMenuSelection (initializationMethod : integer);
var
  calgaryPeople : People;
  noPeople      : integer;
begin
```

James Tam

Putting This All Together (9)

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

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

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

James Tam

Putting This All Together (10)

```
(***** Start of main program*****)
begin
  var initializationMethod : integer;

  displayMenu;
  readln(initializationMethod);
  writeln;
  handleMenuSelection(initializationMethod);
  writeln;
end.
```

James Tam

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

James Tam