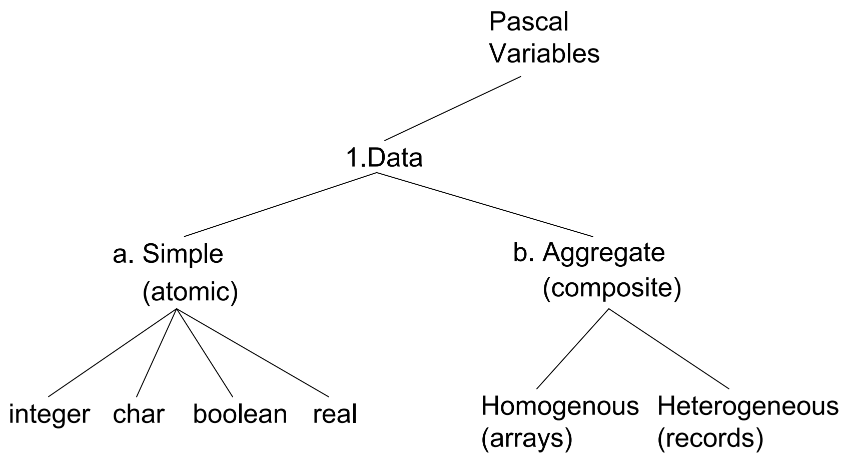
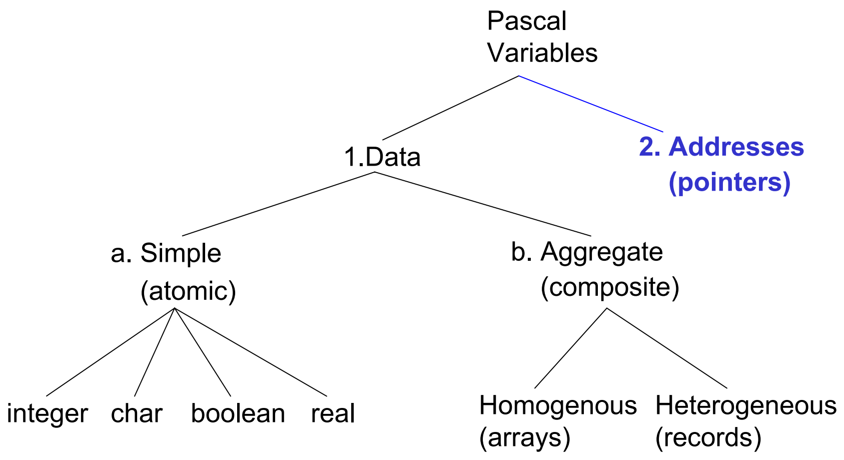
Pointers

In this section of notes you will learn about another type of variable that stores addresses rather than data

Types Of Variables



Types Of Variables



Declaration Of Pointer Variables

Format:

```
type
    type name = ^ type pointed to¹;
        :                  :
begin
    var pointer name : type name;
```

Example:

```
type
    IntegerPointer = ^integer;
        :                  :
begin
    var numPtr1, numPtr2 : IntegerPointer;
```

1 An alternative is to use the "at-sign" @ instead of the "up-arrow" ^ to declare a pointer variable (*not recommended*)

Allocating Memory For Pointers

Static vs. dynamic memory

- arrays

Allocating dynamic memory

- Reserving some dynamic memory and having the pointer point to it.

Format

```
new (pointer name);
```

Example

```
new (numPtr1);
```

De-allocating Memory For Pointers

De-allocating memory

- Returning back the dynamically allocated memory

Format

```
dispose (pointer name);
```

Example

```
dispose (numPtr1);
```

De-allocating Memory For Pointers: Followup

Should also be followed by a statement so that the pointer no longer points to the de-allocated memory.

Format:

pointer name := NIL;

Example

numPtr1 := NIL;

Using Pointers

Important! Are you dealing with the pointer or what the pointer is pointing to (allocated memory)?

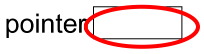
•Pointer name

•Pointer name ^ (de-reference pointer)

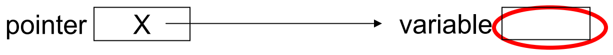
Using Pointers

Important! Are you dealing with the pointer or what the pointer is pointing to (allocated memory)?

•Pointer name



•Pointer name ^ (de-reference pointer)



Accessing Pointers

Format:

(Pointer)

pointer name

(Memory pointed to)

pointer name ^

Example:

(Pointer)

```
writeln(numPtr2);
```

(Memory pointed to)

```
writeln(numPtr1^);
```

Accessing Pointers

Format:

(Pointer)

pointer name

(Memory pointed to)

pointer name ^

Example:

(Pointer)

~~writeln(numPtr2);~~

(Memory pointed to)

writeln(numPtr1^);

Using Pointers : Allowable Operations

Assignment	:=
Relational	
• Equity	=
• Inequality	<>

Using Pointers : Assignment

Format:

(Pointer)

pointer name := *pointer name*;

(Memory pointed to)

pointer name ^ := *expression*;

Example:

(Pointer)

```
numPtr1 := numPtr2;
```

(Memory pointed to)

```
numPtr1^ := 100;
```

Using Pointers : Allowable Operations (Equality)

Format:

(Pointer)

if (*pointer name 1* = *pointer name 2*) then

(Memory pointed to)

if (*pointer name 1* ^ = *pointer name 2* ^) then

Example:

(Pointer)

```
if (numPtr1 := numPtr2) then
```

(Memory pointed to)

```
if (numPtr1 ^ := numPtr2 ^) then
```

Using Pointers : Allowable Operations (Inequality)

Format:

(Pointer)

if (*pointer name 1 <> pointer name 2*) then

(Memory pointed to)

if (*pointer name 1 ^ <> pointer name 2 ^*) then

Example:

(Pointer)

if (numPtr1 <> numPtr2) then

(Memory pointed to)

if (numPtr1 ^ <> numPtr2 ^) then

Pointers : First Example

A full version of this example can be found in Unix under:
/home/231/examples/pointers/pointer1.p

```
program pointer1 (output);

type
   IntegerPointer = ^integer;
          :                         :
begin
   var num, temp              : integer;
   var numPtr1, numPtr2 : IntegerPointer;

   writeln('Example One');
   num := 10;
   new(numPtr1);
   new(numPtr2);
   numPtr1^ := 100;
   numPtr2^ := 100;
```

Pointers : First Example (2)

```
writeln('num = ':11, num:3);
writeln('numPtr1^ = ':11, numPtr1^:3);
writeln('numPtr2^ = ':11, numPtr2^:3);
if (numPtr1 = numPtr2) then
  writeln('numPtr1 and numPtr2 point to the same location in memory')
else
  writeln('numPtr1 and numPtr2 point to two separate locations');
if (numPtr1 ^= numPtr2^) then
  writeln('The data pointed to by numPtr1 and numPtr2 are equal.')
else
  writeln('The data pointed to by numPtr1 and numPtr2 are not equal.');

writeln('Example two');
temp := num;
num := numPtr1^;
writeln('num = ':11, num:3);
writeln('numPtr1^ = ':11, numPtr1^:3);
```

Pointers: First Example (3)

```
writeln('Example three');
numPtr1^ := num;
num := 2;
writeln('num = ':11, num:3);
writeln('numPtr1^ = ':11, numPtr1^:3);

writeln('Example four');
numPtr2 ^ := 66;
numPtr1 := numPtr2;
if (numPtr1 = numPtr2) then
  writeln('numPtr1 and numPtr2 point to the same location in memory')
else
  writeln('numPtr1 and numPtr2 point to two separate locations');
numPtr2^ := 33;
writeln('numPtr1^ = ':11, numPtr1^);
writeln('numPtr2^ = ':11, numPtr2^);
```

Pointers: First Example (4)

```
dispose(numPtr1);
dispose(numPtr2);
numPtr1 := NIL;
numPtr2 := NIL;
```

Pointers As Value Parameters

Need to define a type for the pointer first!

Example (defining type)

```
type
  CharPointer = ^char;
```

Format (passing pointer):

procedure *procedure name* (*pointer name (1)* : *type of pointer (1)*;
pointer name (2) : *type of pointer (1)*;
: :
pointer name (n) : *type of pointer (n)*);

function *function name* (*pointer name (1)* : *type of pointer (1)*;
pointer name (2) : *type of pointer (1)*;
: :
pointer name (n) : *type of pointer (n)*);

Pointers As Value Parameters (2)

Example (passing pointer):

```
procedure proc1 (charPtr : CharPointer);
```

James Tam

Pointers As Variable Parameters

Need to define a type for the pointer first!

Example (defining type)

```
type
  CharPointer = ^char;
```

Format (passing pointer):

procedure *procedure name* (var *pointer name (1)* : *type of pointer (1)*;
var *pointer name (2)* : *type of pointer (1)*;
: :
var *pointer name (n)* : *type of pointer (n)*);

Example (Passing pointer):

```
procedure proc2 (var charPtr : CharPointer);
```

James Tam

Pointers: Second Example

A full version of this program can be found in Unix under:
/home/231/examples/pointers/pointer2.p

```
program pointer2 (output);
type
  CharPointer = ^char;

procedure proc1 (charPtr : CharPointer);
var
  temp    : CharPointer;
begin
  writeln;
  writeln('In procedure proc1');
  new(temp);
  temp^ := 'A';
  charPtr := temp;
  writeln('temp^ = ', temp^);
  writeln('charPtr^ = ', charPtr^);
end;
```

Pointers: Second Example (2)

```
procedure proc2 (var charPtr : CharPointer);
var
  temp : CharPointer;
begin
  writeln;
  writeln('In procedure proc2');
  new(temp);
  temp^ := 'A';
  charPtr := temp;
  writeln('temp^ = ', temp^);
  writeln('charPtr^ = ', charPtr^);
end;
```

Pointers: Second Example (4)

```
begin                  (* Main program *)
  var charPtr : CharPointer;
  new (charPtr);
  charPtr^ := 'a';
  writeln;
  writeln('In the main program.');
  writeln('charPtr^ = ', charPtr^);
  proc1(charPtr);
  writeln('After proc1');
  writeln('charPtr^ = ', charPtr^);
  proc2(charPtr);
  writeln('After proc2');
  writeln('charPtr^ = ', charPtr^);
  writeln;
end.                   (* End of main program *)
```

You Should Now Know

How to declare new types that are pointers to data

How to declare variables that are pointers

The difference between static and dynamically allocated memory

How to dynamically allocate memory

How to de-allocate memory

Why and when to set pointers to NIL

How to access a pointer and how to access what the pointer points to

How to assign values to a pointer and how to assign values to what the pointer points to

What operations can be performed on pointers and how does each one work

How to pass pointers as value and variable parameters