Declarations

Provide information about the name and type of data objects needed during program execution.

Two types of Declarations :-

1) Implicit Declarations
2) Explicit Declarations

① Implicit Declaration or Default Declaration :-

They are those declarations done by Compiler when no explicit declaration or user defined declaration is mentioned.

Example → $abc = 'astring';
$abc = 7;

In 'perl'
↳ Compiler implicitly understand
$abc = 'astring'; — is a string variable
$abc = 7; is an integer variable.

(2) Explicit Declaration –

Float A, B;

float A, B of C lang.

In explicit user explicitly define the variable type.

In this example it specifies that it is of float type of variable which has name A & B

"Declaration" basically serves to indicate the desired life time of data objects.

Declarations of Operations –

Compiler need the signature of a prototype of a subprogram or function so it can determine the type of argument is being used & what will be the result type.

Example, float sub(int x, float y.)

It declares sub to have the signature

sub: int × float → float

operation

Purpose of Declarations :-

(1) Choice of storage Representation → Storage Representation of data types

(2) Storage Management :- for best storage Mgmt.

For Example,

(1) Simple declaration

float A, B;

It tells lifetime of every data objects can be maximum to end of execution time

(2) Runtime Declaration :- C lang and many more lang provide us the feature of dynamic m/m allocation by keywords 'Malloc & Calloc'

So a special block of m/m is allocated in m/m & their lifetime is also different.

Type checking & Type Conversion

Type checking means checking that each operation should receive proper number of arguments and of proper data type

like A = B*j+d;

'*' & '-' are basically int & float data types based operation & if any variable in this A = B*j+d. is of other than int/float then compiler will give generate type error.

Two ways of Type checking

1. Dynamic Type checking
2. Static

(1) Dynamic → (1) Done at runtime

(2) Concept of type tag which is stored in each data objects that indicates the data type of the object

Example → An integer data object contains its 'type' and 'values' attributes

Perl & prolog → Dynamic Type checking

Advantages →

(1) Flexible in designing program.

(2) No declaration are Required

(3) Type may change during execution

(4) programmers are free from concern about data type

Disadvantages –

(1) Difficult to Debug

(2) Extra storage

(3) Hardware support

Static Type checking → Done at Compile time

Information needed at Compile time – by declaration

Information required includes –

(1) for each operation – number, order, data type & its arguments

(2) for each variable → Name & data type of data objects.

Example A+B — in this type of A+B variables must not be changed

(3) For each Constant — Name and data type and value

the value & name is specified & it further if Checked value assigned should be matches datatype

Const int x=28;

Const float x=2.087;

Advantages

(1) Compiles Save Info.

(2) Checked execution paths

Disadvantages — It affects many ~~types~~ Aspects of lang-

(1) Declarations

(2) Data Control Structures

(3) provision of Compiling separately: Some Sub programs

Assignment - A basic operation for changing and binding of a value to a the data object.

- Assignment also returns a value, which is the data object containing a copy of the value assigned

Example → I n c

Assignment (=): integer * integer → integer3

Two concepts through which we can define assignment -

L-value – left value - location for an object

R-value – Right value - Content at that location

A = B [Copying the value of Variable B to variable A]

L-value R-value