Comp151

Definitions & Declarations

Example: Definition

```
/* progam1.cpp */

#include <iostream.h>
#include <string.h>

int global_var = 23;                        // global variable definition

void reverse_print(const char* s)           // function definition
{
    for (int j = strlen(s) - 1; j >= 0; --j)
        cout << s[j];
    cout << endl;
}
```

Example: Declaration

```
/* progam2.cpp */
#include <iostream.h>

extern int global_var;                        // external variable declaration
extern void reverse_print(const char* s);     // external function declaration

void main(int argc, const char* argv[])
{
    float local_var;                          // local variable definition
    local_var = 987.654;

    cout << "global var = " << global_var << endl;
    cout << "local var = " << local_var << endl;
    cout << "input str (backwards) = ";
    reverse_print(argv[1]);
}
```

Definition

- A definition introduces a variable's or a function's <u>name</u> and <u>type</u>.
- A variable definition reserves a number of bytes of memory for the variable.
- A function definition generates code for the function.
- In both cases, definitions cause memory to be allocated to store the variable or function.
- An object must be defined <u>exactly once</u> in a program.

Declaration

- The declaration of a variable announces that the variable exists and is defined somewhere else (in the same file, or in a different file). The connection is made when the object files are linked.
- A declaration consists of the variable's name and its type preceded by the keyword *extern*.
- A declaration does not generate code, and does not reserve memory.
- There can be many declarations for an object name in a program.
- If a declaration is used in a file different from that with the definition of the object, the linker will insert the real memory address of the object instead of the symbolic name.
- In C++, a variable must be defined or declared to the program before it is used.

Advantages of Header Files

In general, a header file provides a centralized location for

- external object declarations
- function declarations
- class definitions
- inline function definitions

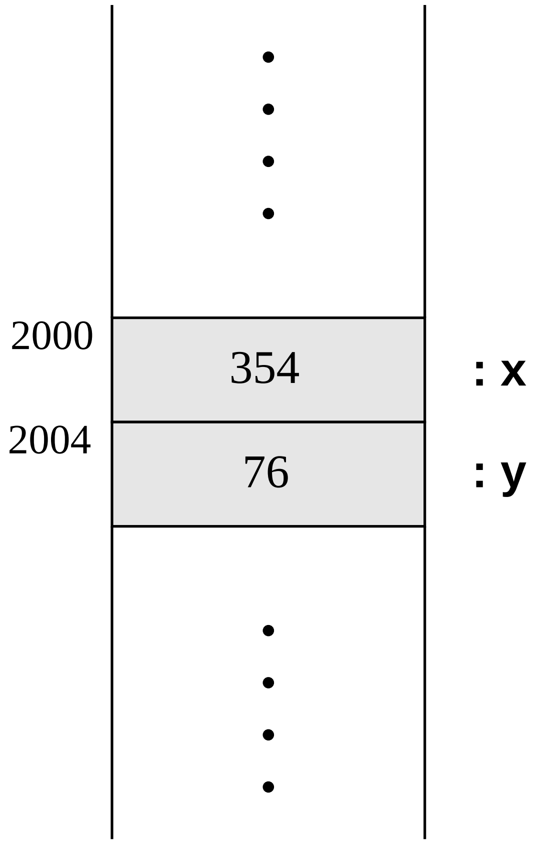
The advantages are:

1. By including the header files, all files of the same piece of software are guaranteed to contain the same declaration for a global object or function.
2. Should a declaration require updating, only one change to the header file will need to be made.

Variables

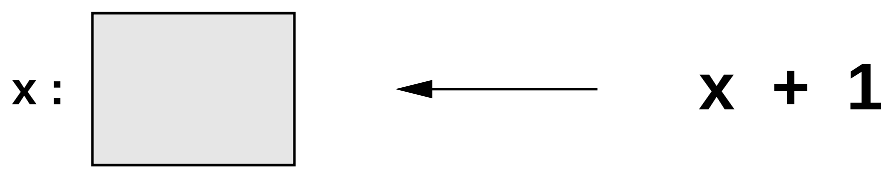
A variable is a symbolic name assigned to some memory storage.

- The size of this storage depends on the type of the variable. e.g. char is 1-byte long and int is 4-byte long.
- The difference between a variable and a literal constant is that a variable is addressable.



lvalue & rvalue

[interpretation of " x = x + 1 "]



A variable has dual roles, depending on where it appears in the program, it can represent

- lvalue: the location of the memory storage
- rvalue: the value in the storage

They are so called because a variable represents an lvalue (a rvalue) if it is written to the *left (right)* of an assignment statement.

Thus, the following are invalid statements in C++:

```
4 = 1;
grade + 10 = new_grade;
```