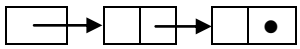
Names and Scope Binding

Static Binding: Fast

- Language Design Time Binding, e.g. Reserved Keywords.
- Language Implementation Time Binding, e.g. Bit Allocation to different types and Stack & Heap Size.
- Compile-Time Binding, e.g. Constant Values, Function(s) in the same file.
  C/C++:
  - #define TRUE 1
  - #define FALSE 0
- Link-Time Binding, e.g. Call to a function in another file.

Dynamic Binding: Slow, Flexible

- Load-Time Binding, e.g. Assignment of a physical memory.
- Run-Time Binding, e.g. Allocation in heap (new and delete).



Scope

- Scope governs the visibility if the bindings
- Reference Environments store bindings and map names to the attributes.

```
#include <iostream>
using namespace std;

int main() {
    int x=1;
    cout << x << endl;

    if( 1==1 ) {
        int x=2;
        cout << x << endl;
    }

    cout << x << endl;
    return 0;
}
```

C/C++ Output: 1 2 1
Java refuses to compile!!!
Java Script Output: 1 2 2 (Uses the latest scope)

Different Languages have different rules.

- Nested Scope (Sub-Scope) Allowed?
- Global Scope?

Task: Test all routines given in the document “Scope and Binding” using any language of your choice and determine the basic scope rule(s).

```
int x;
void f(int m) {
      float x,y;
      …..
      {
            int i,j;
            float u,v;
            …..
      }
      …..
}

int g(int n) {
      bool t;
      …..
}
```