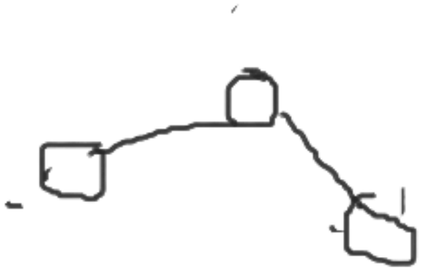
Symbol Table

1. Variable
2. Function
3. Data type
4. Object
5. Class
6. Interface

Symbol Table implementation



1. List (sorted & unsorted)
2. Hash Table
3. Binary Search Tree

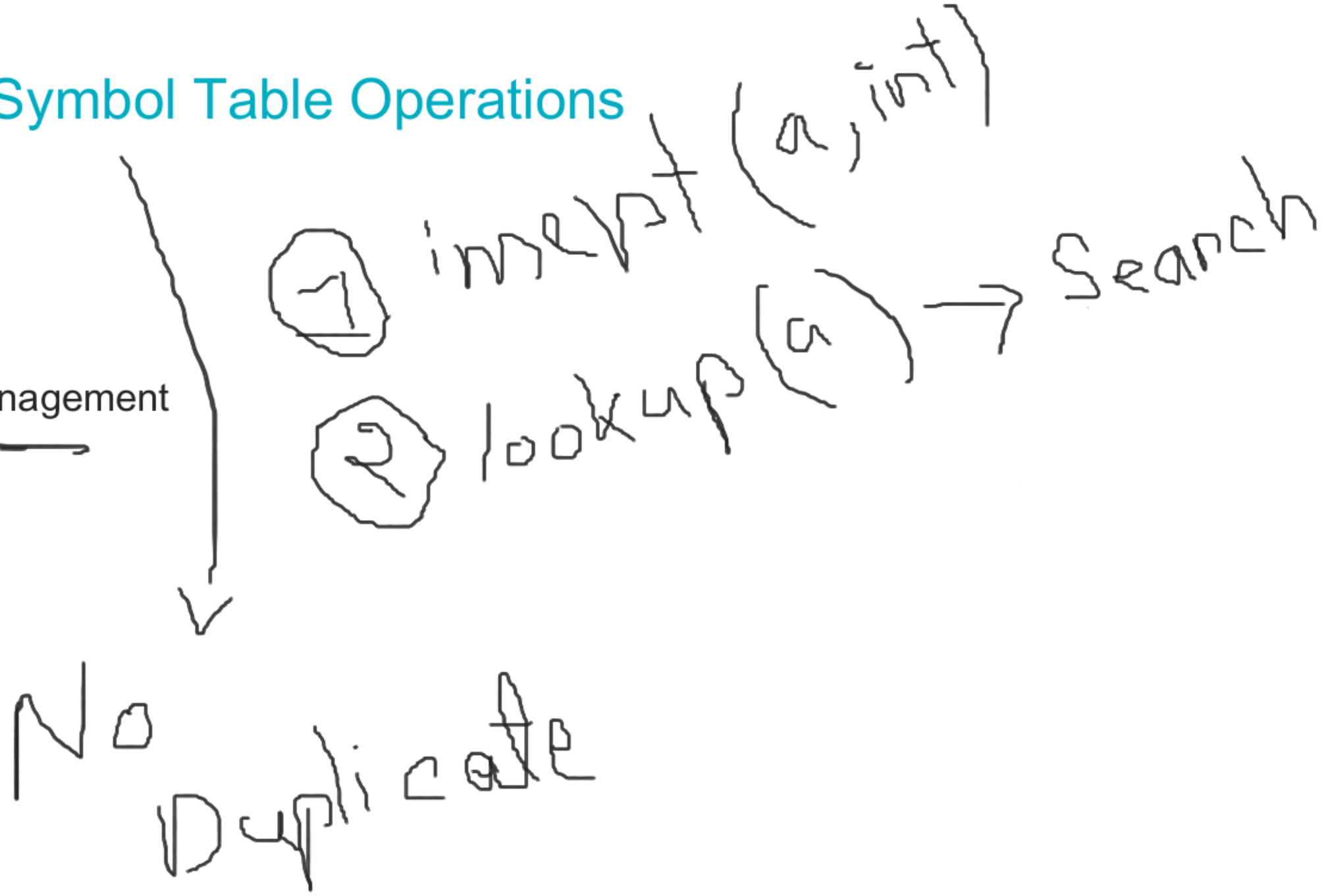
Symbole Table

```
1 main ()
  {
2 char a [5 ] ;
3 int x
4 x = 5
5 print (x)
```

Name	Type	Size	Dimension	Line of Declaration	Line of Usage	Address
a	char	5	1	2	—	memory location
x	int	bit	0	3	4 → 5	memory location

Symbol Table Operations

1. Insert ( )
2. lookup ( )
3. Delete ( )
4. Scope Management

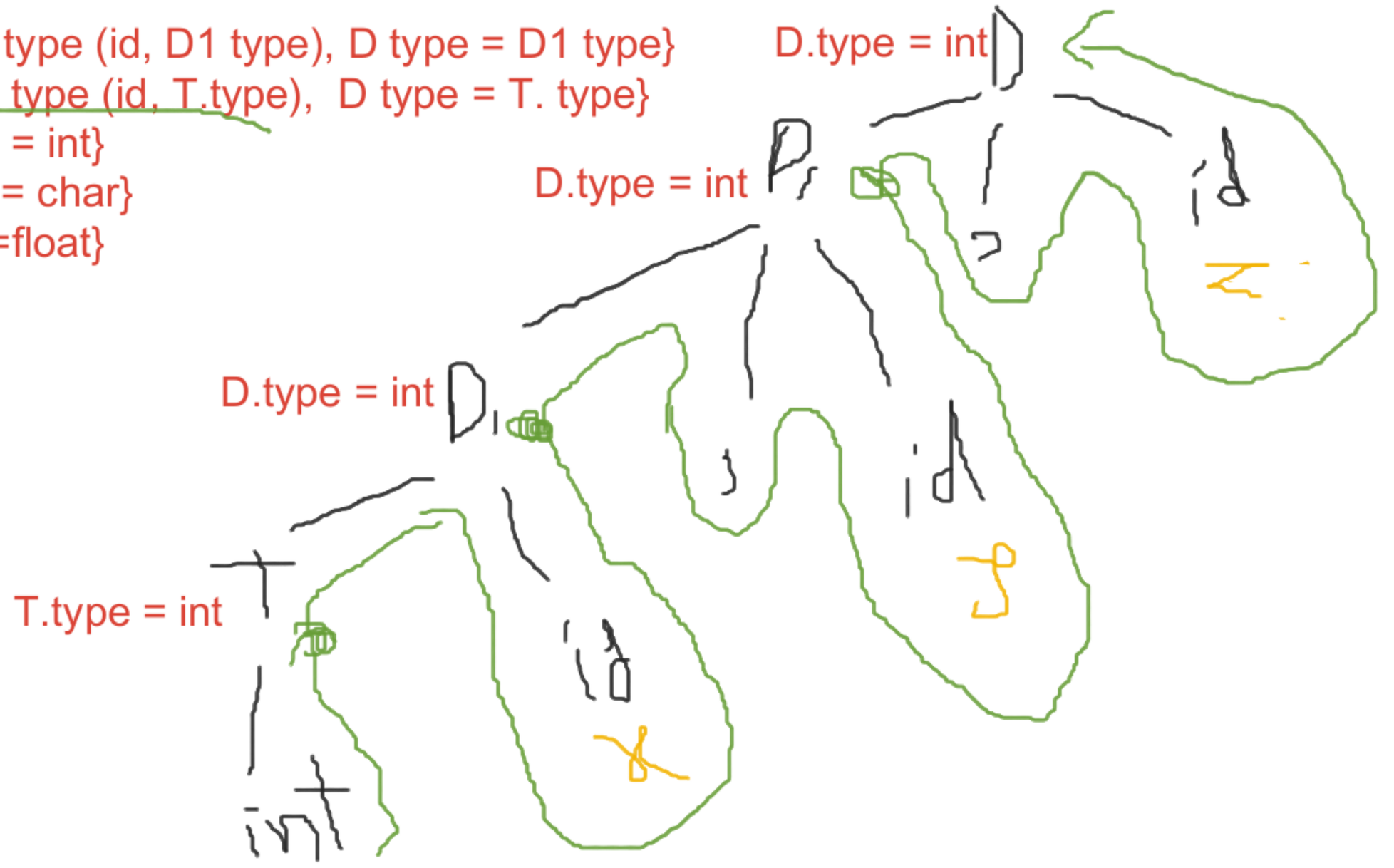


SDD to Store Information Type

D --> D1, id {Add type (id, D1 type), D type = D1 type}
D --> T id {Add type (id, T.type), D type = T. type}
T --> int {T.type = int}
T -->char {T.type = char}
T -->float {T.type =float}

input : int x, y , z

v. name	Data type
x	int
y	int
z	int



Scope management

```
int value;
void one ( )
{
    int a;
    {
        int b;
    }
    int c;
    {
        int  d;
    }
}
Void two ( )
{
    int e;
}
```

Global Symbols		
Name	Type	Data Type
Value	variable	int
one	Function	void
two	Function	void

Function One		
name	Type	Data type
d	variable	int
c	variable	int

Function Two		
name	Type	Data type
e	variable	int

b	variable	int

d	variable	int

Scope management

```
int x
int y
void sum (int m){
    float x,y
     if(-----){
         int i,j
     }
     else {
         double i,j
     }
}

int g (int n){
    bool flag;
}
```