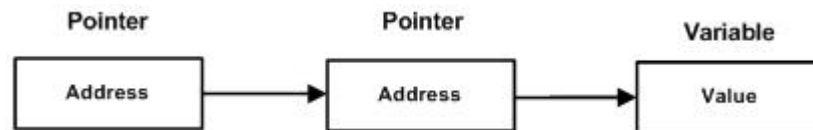


# C - Pointer to Pointer

A pointer to a pointer is a form of multiple indirection, or a chain of pointers. Normally, a pointer contains the address of a variable. When we define a pointer to a pointer, the first pointer contains the address of the second pointer, which points to the location that contains the actual value as shown below.



A variable that is a pointer to a pointer must be declared as such. This is done by placing an additional asterisk in front of its name. For example, the following declaration declares a pointer to a pointer of type int -

```
int **var;
```

When a target value is indirectly pointed to by a pointer to a pointer, accessing that value requires that the asterisk operator be applied twice, as is shown below in the example -

[Live Demo](#)

```
#include <stdio.h>

int main () {

    int  var;
    int  *ptr;
    int  **pptr;

    var = 3000;

    /* take the address of var */
    ptr = &var;

    /* take the address of ptr using address of operator & */
    pptr = &ptr;

    /* take the value using pptr */
    printf("Value of var = %d\n", var );
```

```
    printf("Value available at *ptr = %d\n", *ptr );  
    printf("Value available at **pptr = %d\n", **pptr);  
  
    return 0;  
}
```

When the above code is compiled and executed, it produces the following result -

```
Value of var = 3000  
Value available at *ptr = 3000  
Value available at **pptr = 3000
```

---

---