CHAPTER 8

Arrays

Why arrays are needed?

Ex: To add three numbers.

```
int a ,b,c,d;
a=b+c+d;                    works
```

To add four numbers.

```
int a,b,c,d,e
a=b+c+d+e;                  works
```

suppose to add 200 numbers 2000 numbers

Problem ! Unique symbolic reference names.

1) We need to come up with so many symbolic reference names.
2) Symbolic table is too large.
3) Code is very long.

So arrays are used with single symbolic reference name.

Consecutive addresses needed to access any one item

Note:-You know one,you know all.

Numbering of the elements : Starts with zero like floors of a building.

Array declarations:-

```
int  total ;
int  name[3];
```

Symbol table

Total	int	2000
name	int	2002

Note: Array elements in Consecutive address of zeroth element given in symbol table.

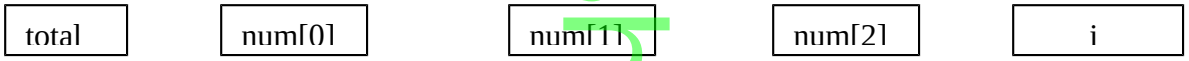
Data access

Total : contents of(2000)
num[0] :contents of(2002)
num[1] : contents of(2004)
num[2] :contents of(2006)

"[ ]" is called a binary unix operator.

Memory map

Ex: int total : 2000
int num[3]:2002-6
int i: 2008



2000 2002 2004 2006 2008

Out of bond access

C allows for some special cases like the above.

Ex: num[-1] access total's value
num[3] access i's value

Why OOB access? If porgs have bond checking they occupy size and takes more time to execute bond checking code.

Types of array: (4 types)

1)static 2)dynamic

Dimensions:
1)Single dimensional array int y[10];
2) Two dimensional array int z[4][5];

Array Rule

y[2.5] access y[2]
y[10.9] access y[10]

Truncation is done here
Y[2] , y[-1] , y[x] ,y[y[x]] are valid where x is int.

Covery rule

y[y[x]]

Contents of (2002+2*contentsof(2000))
Hence: y[y[x]] is contents of (2002+2*contents of(2002*(contents of(2000)))

Therefore ,
Y[0]:contents of (2002+2*0)
Y[1]:contents of (2002+2*1)
Y[2]:contents of (2002+2*2)
Consecutive bytes allocates.