Pointer to pointer

- The pointer that stores address of another pointer variable is called as ‘pointer to pointer’.
- Example :-

```
int x=10;   int *px=&x;
int **ppx=&px;
printf(“%d”,**ppx);
```

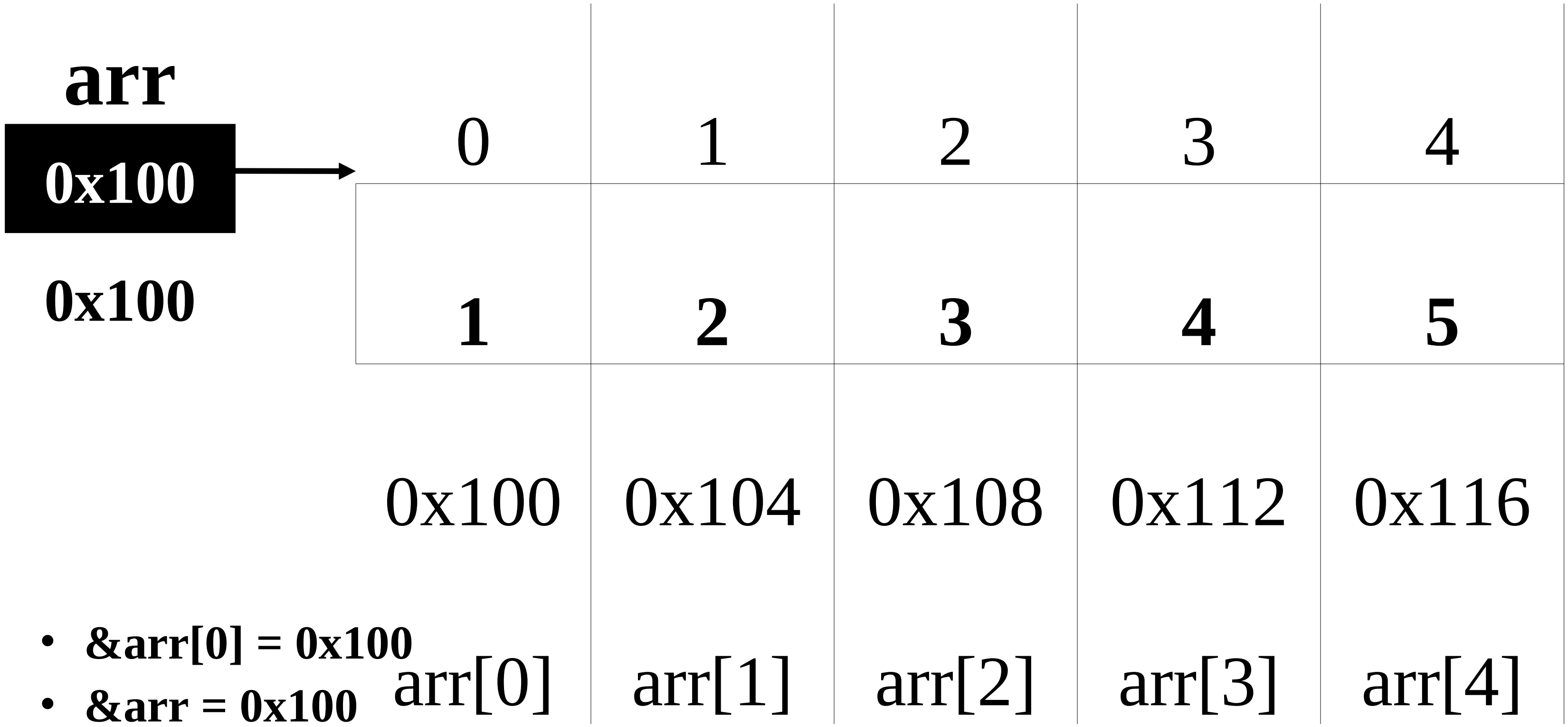
Array

- Array is collection of similar data elements in contiguous memory locations.
- Elements of array share the same name i.e. name of the array.
- And they are identified by unique index in the array called as subscript.
- Array indexing starts from 0.
- Checking array bounds should be done by programmer.

1D Array - Syntax

```
int main(){
   int arr[5] = { 1, 2, 3, 4, 5 };
   int index;
   for(index=0;index<5;index++)
   printf(“\n%d”,arr[index]);
   printf(“size = %d”,sizeof(arr));
}
```

1D Array - Fig



arr

0x100

0x100

0	1	2	3	4
1	**2**	**3**	**4**	**5**
0x100	0x104	0x108	0x112	0x116
arr[0]	arr[1]	arr[2]	arr[3]	arr[4]

- **&arr[0] = 0x100**
- **&arr = 0x100**
- **arr = 0x100**
- ***arr = 1**

1D Array - Syntax

```
int main() {
    int arr[5], index;
    printf(“Enter 5 elements..”);
    for(index=0;index<5;index++)
        scanf(“%d”,&arr[index]);
    for(index=0;index<5;index++)
        printf(“\n%d”,arr[index]);
}
```

1D Array & pointers - Syntax

```
int main() {
    int arr[5], *ptr, index;
    ptr = arr ;
    printf(“Enter 5 elements..”);
    for(index=0;index<5;index++)   {
         scanf(“%d”, ptr);
         ptr++;
    }
    ptr = arr;
    for(index=0;index<5; index++, ptr++)
          printf(“\n%d”,*ptr);
}
```

Array & pointer

- int arr[5] ;

arr[index] is internally resolved as *(arr+index)

```
arr[index] == *(arr+index)
 *(index+arr) == index[arr]
```

- int *ptr=arr;

```
*(ptr+index) == ptr[index]
*(index+ptr) == index[ptr]
```

Passing array to function

- Arrays are passed to function by ref.
- The address of starting element of array (Base address) is passed to the function.
- The address is collected in a pointer.
- We can use pointer as well as array notation to access array elements in the function.