C Language

Application of Pointers

Saurabh Shukla (MySirG)

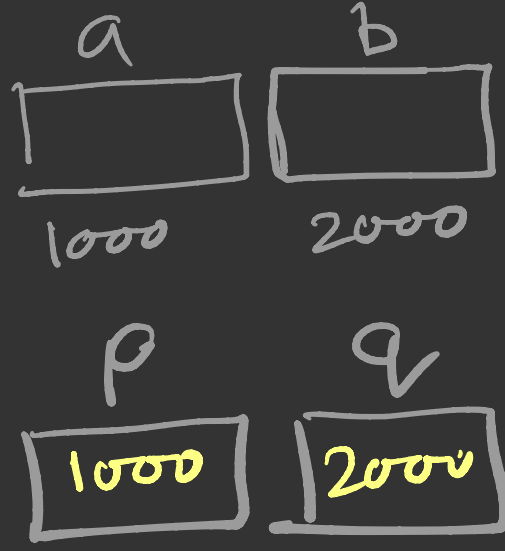
Agenda

1. Pointer's Arithmetic
2. Call by reference
3. Pointers and arrays
4. Pointers and strings
5. Array of pointers
6. Pointer to array
7. Wild pointer
8. NULL pointer
9. Dangling pointer
10. Void pointer

- pointer is a variable
- It contains address of another variable
- size of pointer is not dependent on its type.
- Pointer jis type ka hota usi type ke variable ko point karta hai
- int *p;
- *p ⇝ variable pointed by p

Pointer's Arithmetic

```
int a, b, *p, *q;
p = &a;
q = &b;
```



q - p 250

- p + q
- p * q
- p / q
- p * 5
- p / 4

} Wrong

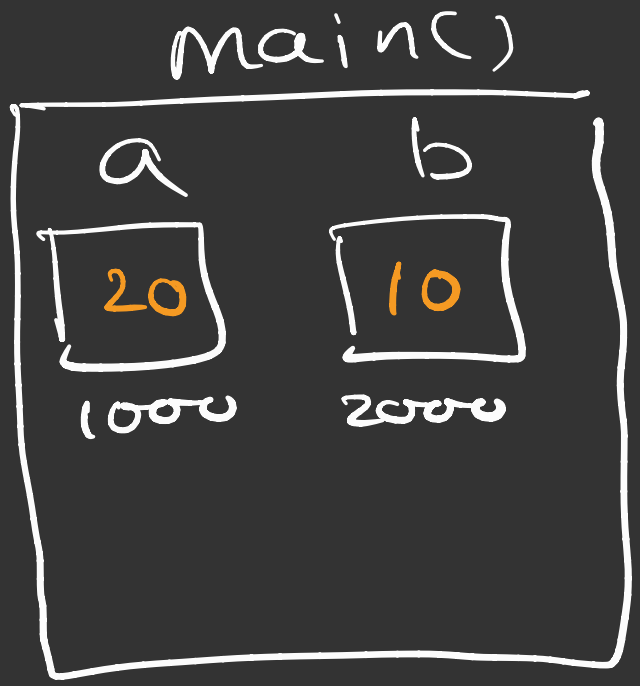
p + 1	1004	p - 2	992
p + 2	1008		
q + 5	2020		

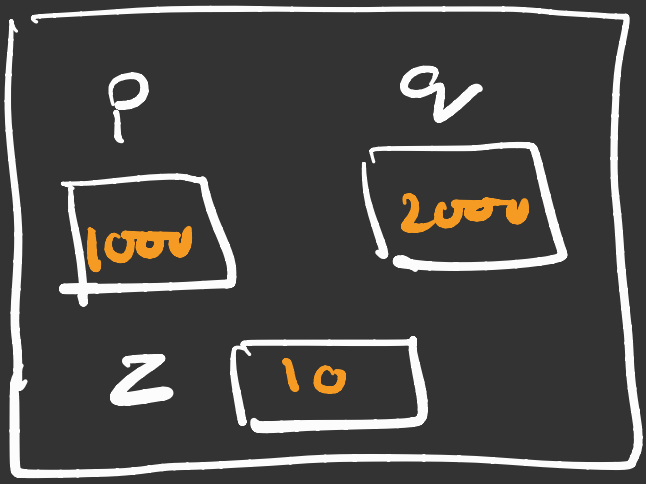
pointer + K

address in pointer + K * (sizeof type of pointer)

2000 + 5 * sizeof (int)

```
void swap(int *, int *);
int main()
{
    int a, b;
    printf("Enter two numbers");
    scanf("%d %d", &a, &b);
    swap(&a, &b);
    printf("%d %d", a, b);
    return 0;
}
void swap(int *p, int *q)
{
    int z;
    z = *p;
    *p = *q;
    *q = z;
}
```





Call by Reference

```
void swap(int *, int*);
int main()
{
    int a,b;
    printf("Enter two numbers");
    scanf("%d %d",&a,&b);
    swap(&a,&b);
    printf("%d %d", a, b);
    return 0;
}
void swap(int *p, int *q)
{
    int t;
    t = *p;
   *p = *q;
   *q = t;
}
```

call by reference

Formal argument

1. ordinary variable
2. pointer variable

Why we use & in scanf() ?

```
scanf("%d", &x);
```