

C Programming

LECTURE 9

C Programming

- Pointer
 - Each variable in the code is mapped to an appropriate location in RAM
 - One can find this address by placing an '&' in front of the variable name

C Programming

- You can find out the memory location (address) that holds the value of a given variable by placing an **'&'**

```
#include <stdio.h>
int main()
{
int a=10;
printf("a=%d\n", a);
printf("Address of a=%d\n", &a);
return 0;
}
```

- example

C Programming

- **A pointer is a variable.** It holds the address of another variable

```
#include <stdio.h>
int main()
{
float a=20.0;
float *pa;
pa=&a;
printf("%d\n", pa);
printf("%d\n", &a);
return 0;
}
```

Note: `float *pa` declares `pa` as a pointer. It declares type for variable 'a'.
The pointer itself is always an integer

example

C Programming

```
#include <stdio.h>
int main()
{
float a =20.0, b=50.0;
float *pa, *pb;
pa=&a; pb=&b;
return 0;
}
```

example

C Programming

- **Sometimes within the program you may want to examine what values the pointer actually refers to**

```
#include <stdio.h>
int main()
{
float a=20.0;
float *pa; pa=&a;
printf("%f\n", *pa);
printf("%f\n", a);
return 0;
}
```

Note: **pa* gives the same value as *a*

C Programming

- **One reason to use Pointer**

```
#include <stdio.h>
void tentimes(float a)
{
    a = 10.0*a;
}
int main()
{
    float a=20;
    tentimes(a);
    printf(" a = %f\n", a);
    return 0;
}
```

example

```
#include <stdio.h>
void tentimes(float *pa)
{
    *pa = 10.0* *pa;
}
int main()
{
    float a=20;
    tentimes(&a);
    printf(" a = %f\n", a);
    return 0;
}
```

example

C Programming

- **Following usage of pointer is incorrect**

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
int a=3, b=2, *pa=&a, *pb=&b;
```

```
&b=pa;
```

```
pa=a;
```

```
*pa = pb;
```

```
pa=*pb;
```

```
return 0;
```

```
}
```

Address of variable 'b' cannot be changed
pa is a pointer, a is not
Variables of different kind

C Programming

- **Pointers and arrays**

- **an array is like a pointer**

```
#include <stdio.h>
int main()
{
float a[3]={1.0, 2.0, 3.0};
printf("a=%d",a);
return 0;
}
```

- **an array "a" is actually a pointer to the 0-th element of the array while a[0], a[1] and a[2] act just like regular variables.**
- **The content of a is the address which points to a[0].**

example

Numerical Analysis

- Since it is a pointer

Array access	Pointer equivalent
a[0]	*a
a[1]	*(a+1)
a[2]	*(a+2)

Example 9-6

C Programming

- Since an array is like a pointer, we can pass an array to a function, and modify elements of that array. (Example []) and (Example *)
- A function can take an array as an argument and can modify the contents of the array.