- ✔What Are Pointers
- ✔Pointer Variables
- ✔Pointer Operators
- ✔Pointer Expressions
  - ✔Pointer Assignments
  - ✔Pointer Arithmetic

Pointers

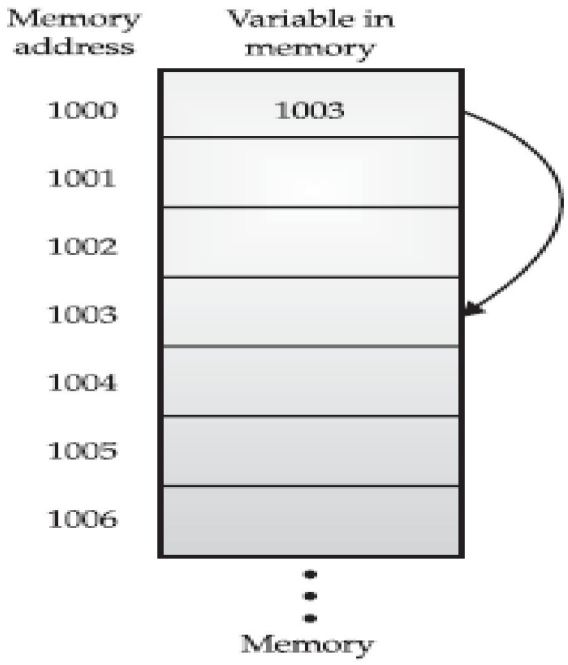
Part-1

For Detailed Reading, Consult
The Complete Reference C++, Chapter-5

What Are Pointers

- A *pointer* is a variable that holds a memory address.
- This address is the location of another object (typically another variable) in memory.

  - For Example



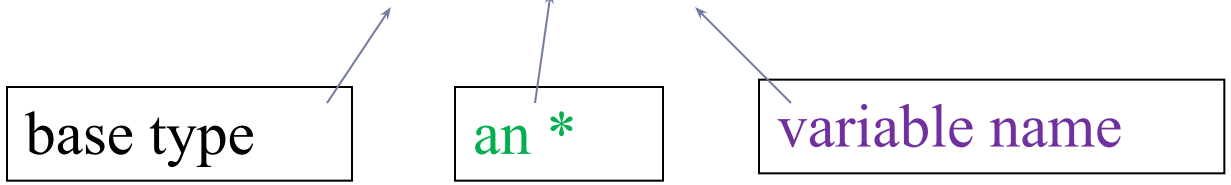
Pointer Variables

- If a variable is going to hold a pointer, it must be declared as such

- A pointer declaration consists of
  - a base type
  - an *
  - and the variable name

*type *name;*

For Example `int *ptr;`

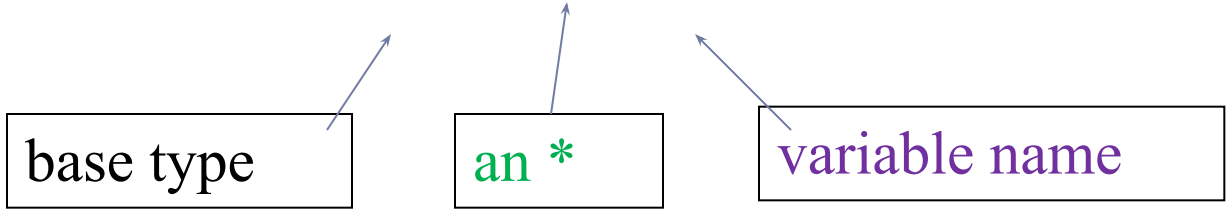


Pointer Variables Cont.

- The base type of the pointer defines what type of variables the pointer can point to

- Technically, any type of pointer can point anywhere in memory.
- All pointer arithmetic is done relative to its base type

Example `int *ptr;`



The Pointer Operators

- There are two special pointer operators:
  - **&**
  - *
- The **&** is a unary operator that returns the memory address of its operand

For Example `m = &count;`

places into **m** the memory address of the variable **count**

It has nothing to do with the value of **count**

The Pointer Operators Cont.

- There are two special pointer operators:
  - **&**
  - ***
- Operator * is the complement of **&**.
- It is a unary operator that returns the value located at the address that follows
- For Example if **m** contains the memory address of the variable **count**

`q = *m;` places the value of **count** into **q**

The Pointer Operators Cont.

- There are two special pointer operators:
  - **&**
  - *

Both **&** and * have a higher precedence than all other arithmetic operators except the unary minus, with which they are equal

Pointer Expressions

Pointer Assignments

- As with any variable, you may use a pointer on the right-hand side of an assignment statement to assign its value to another pointer

For Example

```
#include <stdio.h>
int main(void)
{
    int x;
    int *p1, *p2;
    p1 = &x;
    p2 = p1;
    cout<< p2;  /* print the address of x, not x's value! */
    return 0;
}
```

Both **p1** and **p2** now point to **x**

Pointer Expressions

Pointer Arithmetic

- There are only two arithmetic operations that you may use on pointers:
  - **Addition**
  - Subtraction

For Example let **p1** be an integer pointer with a current value of 2000

```
p1++;
```

Causes **p1** to have the value 2002

The reason for this is that each time **p1** is incremented, it will point to the next integer

Assume integers are 2 bytes long

Pointer Expressions

Pointer Arithmetic Cont.

- There are only two arithmetic operations that you may use on pointers:
  - Addition
  - **Subtraction**

For Example let **p1** be an integer pointer with a current value of 2000

```
p1--;
```

Causes **p1** to have the value 1998

The same reason is for subtraction

Assume integers are 2 bytes long

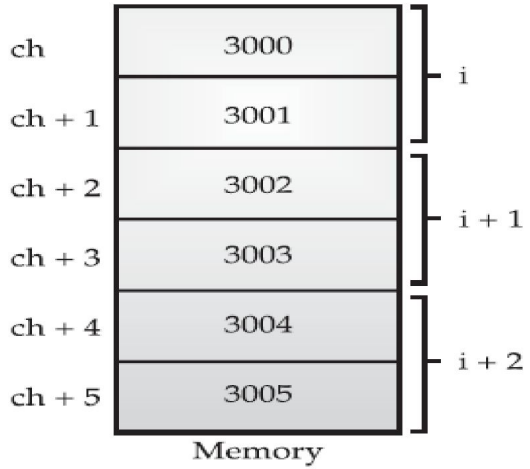
Pointer Expressions
Pointer Arithmetic Cont.

- There are only two arithmetic operations that you may use on pointers:
  - Addition
  - Subtraction

- When applied to character pointers, this will appear as "normal" arithmetic because characters are 1 byte long.

Example

```
char *ch=(char *)3000;
int* i=(int*)3000;
```



Pointer Expressions

Pointer Arithmetic Cont.

- There are only two arithmetic operations that you may use on pointers:
  - Addition
  - Subtraction

- You are not limited to the increment and decrement operators.

For example, you may add or subtract integers to or from pointers.

The expression `p1 = p1 + 12;`

makes **p1** point to the twelfth element of **p1**'s type beyond the one it currently points to.

Pointer Expressions

Pointer Arithmetic Cont.

- There are only two arithmetic operations that you may use on pointers:
  - Addition
  - Subtraction

- You may subtract one pointer from another in order to find the number of objects of their base type that separate the two.

- All other arithmetic operations are prohibited.
  - you may not multiply or divide pointers
  - you may not add two pointers
  - you may not apply the bitwise operators to them