**Chapter10: Arrays**

-Type array\_name[array\_size]

-To access element

-Arrays are sequences of objects of the \_ same type

-First element has index and last element has index # element - 1

-Can declare with initialization: type array\_name [element]

**Chapter11: Pointers**

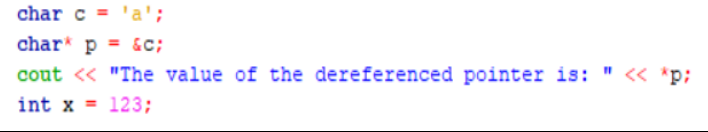
- type \*pointer\_name

- Hold another variables

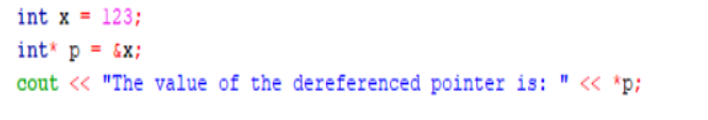
-&x //returns the address of variable x

-\*p //returns the value at the address p

-p = &x; //assigns p the address of variable x



Output => a



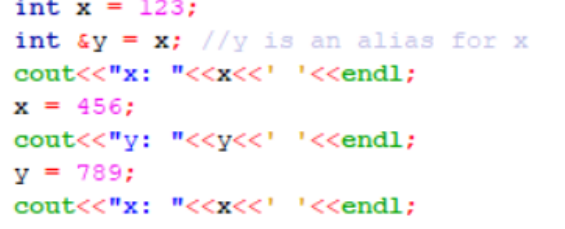
Output => 123

**Chapter12: References**

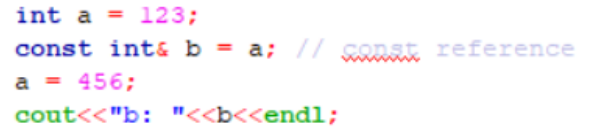
- type &ref\_name

- An alias to an existing variable in memory.

- reference have to be initialized at declaration



Output => x:123 y:456 z:789



Output => b:456

**Chapter 13:Strings**

- Include in C++ standard library

- Create string s: std::string s = text

- Add two strings new\_string = string\_1 + string\_2

- Compare string: ==

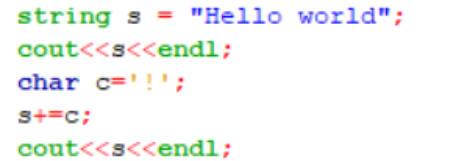
- Input string: std::cin or std::getline

- Pointer to string: .c\_str() member function

- Substring:s.substr(starting\_position, length)

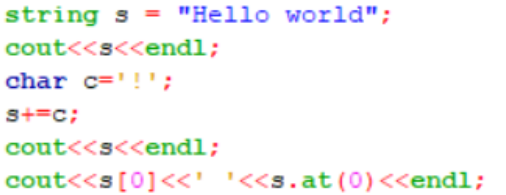


-Output => Hello world!



-Output => Hello world

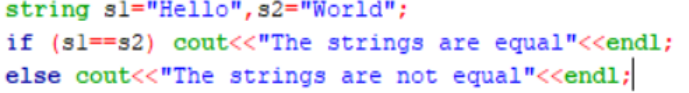
Hello world!



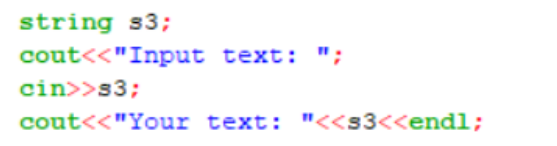
-Output => Hello world

Hello world !

H H

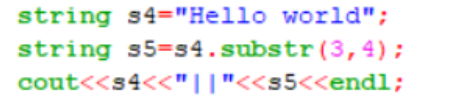


-Output => The strings are not equal

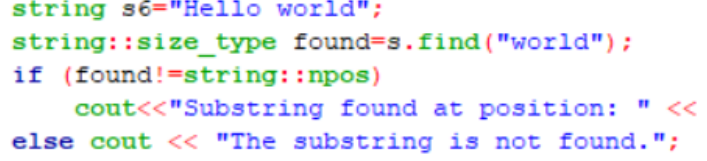


- Output => Input text :Abc

Your text: Abc



-Output => Hello world|| lo w



-Output => Substring at postiton

**Chapter 14: Automatic type deduction**

- Deduce type of object automatically

- Syntax: auto x = 123; //or “hello”, or 123.45,…

- Use when deduce some the type ourselves.