PLAYING WITH CHARACTER DATA TYPR

// Example program

#include <iostream>

using namespace std;

int main()

{

char c = 'abc';

cout<< c;

return 0;

}

THIS OUTPUTS c as

c

The last character of the initialized string.

However in the case of a cin statement, or input from the user:

Cin<<a;

Cin<<b;

Cout<< a;

Cout<< b;

Cin<<c;

Cout<< c;

IF the input is abcd:

Output will be

A will be assigned to a

B will be assigned to b

C will be assigned to c

D will not assigned to anything, unless another cin is given for another variable to which d will be assigned.

STRINGS

\

Inputting strings will not take anything after a space

// Example program

#include <iostream>

using namespace std;

#include <string>;

int main()

{

string c,d;

cin >> c;

cout <<c;

cin>>d;

cout <<d;

return 0;

}

If inp§ut is helloworld

This will assign hello to c and world to d.

However, if initialized within the program;

// Example program

#include <iostream>

using namespace std;

#include <string>;

int main()

{

string c = "Hello World";

cout<< c;

return 0;

}

Will OUTPUT

Hello World

for(int i=1; i<=10;i++)

* The **init** step is executed first, and only once. This step allows you to declare and initialize any loop control variables. You are not required to put a statement here, as long as a semicolon appears.
* Next, the **condition** is evaluated. If it is true, the body of the loop is executed. If it is false, the body of the loop does not execute and the flow of control jumps to the next statement just after the 'for' loop.
* After the body of the 'for' loop executes, the flow of control jumps back up to the **increment** statement. This statement allows you to update any loop control variables. This statement can be left blank, as long as a semicolon appears after the condition.
* The condition is now evaluated again. If it is true, the loop executes and the process repeats itself (body of loop, then increment step, and then again condition). After the condition becomes false, the 'for' loop terminates.

This is why:

For cout<< i

We have 11 for both

for(int i=1; i<=10;i++)

for(int i=1; i<=10;++i)

Arrays

How memory is allocated:

#include <iostream>

using namespace std;

int main()

{

double score[5];

double grade;

for (int i = 0; i < 5; i++)

{

cout << "Enter grade:";

cin >> grade;

score[i] = grade;

}

for (int i = 0; i < 5; i++)

{

cout << &score[i] << endl;

}

cout << sizeof(score) << endl;

return 0;

}

Output

Enter grade:1

Enter grade:2

Enter grade:3

Enter grade:4

Enter grade:5

001CF930

001CF938

001CF940

001CF948

001CF950

40

Press any key to continue . . .

Sizeof(arr)= gives= 40

No of elements is = 5

Therefore, step= 40/5= 8

So you can find address of all indices from first index