

Forum:[Hack Forums](#)

Thread: [Basic C++ tutorial](#)

hacku4eva - July 23, 2010, 5:32 pm

Basic C++ tutorial

In this tutorial I'm going to go over the basics of C++ for all beginners that want to learn the language.

OK let's get started! I'm assuming that you already have Visual C++ download (or any other one that you may have will just do) if you don't have C++ installed already then please click [here](#) and install it so we can begin.

OK, now that we have C++ installed we're ready to start the tutorial...launch VC++ then select "New Project" --> "Win32 Console Application" then name your project (whichever name you prefer) then it'll load another form, click the "Next" button (don't click finish right away) then select "Empty Project" and finally click finish ?

Now for the fun part

===== you'll notice after you have clicked the finish button that the screen looks blue and there isn't a place to start coding, we'll have to add a ".cpp" file to start and in order to add that file you'll need to click on: Solution Explorer --> right click on "Source File" --> "Add.." --> "New items.." then select C++ file (.cpp), name it and click add.

Pre-Processor header(s)

=====

the Pre-processor headers are always your step zero in coding, you'll need to add those in order to use the built in functions that are in c++.

the Pre-processor header you'll see the most in your coding as you start learning C++ is

Code `#include <iostream>` this pre-processor header lets you access the built in function to input and output things from the command screen..like so

Code `#include <iostream>`

using namespace std; // std stands for standard, we'll be adding it under all the pre-processor headers in //every example for ease.

```
int main()
```

```
{
```

```
    cout << "Hello World";
```

```
    return 0;
```

```
}when using the " cout << " operator it basically means the the console will output whatever you typed in between the parenthesis after the operator and it print it to the user sitting in front of the PC.
```

*we'll talk about how to input from users sitting in front of the PC in the next Pre-processor header.

The next Pre-processor header we'll learn about is

Code `#include <string>`

this one is fairly easy, it lets you store words "strings" and use them through out your program, here's an example..

Code `#include <iostream>`

```
#include <string>
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    int age;    // this declares a integer called "num"
```

```
    string name;
```

```
    name = "hacku4eva"; // you could have aded this at the line above if you wish but i prefer coding  
that way
```

```
    cout << "Is your name " << name << "?";
```

```
    cout << endl;
```

```
    cout << " please enter your age ";
```

```
        cin >> age;
```

```
    [cout << endl;
```

```
    cout << " Oh so your name is " << name << " and you are " << age << " years old";
```

```
    return 0;
```

```
} Now lets talk about the new things we haven't talked about that are included in this code
```

A) we have "int age;" that basically tells you that this part will only accept an integer number and it'll be called "age;"

B) we defined a string by calling it "name" and we inputed a word in the following line of it.

C) " cout << endl " that is being used from the first Pre-processor header we talked about and it simply just starts a new line.

D) " cin >> age" cin stands for console-output and what that does is it fetches data from the user that's sitting on the PC

lets stop with those two for now ?

```
=====
```

If/else statements & switch statements

```
=====
```

if/else statement are used to check statements either if they're [equivalent, not equivalent, less than/ greater than, and/or, less than or equal to or greater than or equal to] here's an example..

Code if (day == 3)

```
    cout << " too late..";
```

```
else
```

```
    if ( day == 2 || day == 1 )
```

```
        cout << " you're finally here";
```

```
else
```

cout << " just go home";where it says Code if (day == 2 || day == 3) means either if day is equivalent to 2 OR 1 print out this...

Now to see that same exact code using switch statements

```
=====
```

Code switch (day)

```
{
```

```
    case 3:
```

```
        cout << " too late..";
```

```
        break;
```

```
    case 2:
```

case 1:

```
cout << " you're finally here";  
break;
```

default:

```
cout << "just go home";
```

break; if you looked after case 3 you'll see that right under the cout statement there is something called "break" and what that does is that it basically separates each case from the other. *but if you look at case 2 you'll notice that there isn't a break under it so it goes on to the next statement which is case one so if its either case 2 OR case 1 it'll print out whatever is there.

the default is used when there isn't anything its comparing to (look at the if/else statement, its just like the " else " " //code "

that's all for that, i hope its kinda clear and if it isn't feel free to ask ?

=====

Functions

=====

so far if you looked at the first few codes you'll see that we've been doing all the coding in the main function but as you learn to code and get better at it dumping all your codes in the main function will seem kinda sloppy, thats where functions come to help ?

there are two parts in functions that i need to talk about which are:

- A) value returning functions such as: int, float, double, char, bool (Boolean is always true or false)
- B) the other is non-value returning function which is: void

i'll show a value returning function using a int

Code #include <iostream>

#include <string>

using namespace std;

```
int test();
```

```
int main ()
```

```
{  
    cout << "Hi" << endl;  
    test();  
    return 0;  
}
```

```
int test()
```

```
{  
    cout << " this is a test" ;  
    return 0;  
}
```

if you look on top of the main function you'll see Code int test(); this is called a prototype and is used to call other functions because if you didn't include prototypes the compiler while run through the main function and it will hit the Code test(); and wont know what that is because you didn't tell it to go and get that function at first.

I'll use the same exact code but using a void (non-value returning function)

Code #include <iostream>

#include <string>

```
using namespace std;
```

```
void test();
```

```
int main ()  
{  
    cout << "Hi" << endl;  
    test();  
    return 0;  
}
```

```
void test()  
{  
    cout << " this is a test" ;
```

```
}as you can see that when using the void function you don't need to include "return"
```

that will be all for now, please comment and if I've made any mistakes in here tell me and I'll fix it and of coarse if there isn't anything that you didn't get feel free to ask and I'll try to explain it differently

hope someone finds it helpful ?

Hyde - July 23, 2010, 6:12 pm

Many posts like this but, Thanks for the share.

I can see you put effort into it.

hacku4eva - July 23, 2010, 6:16 pm

(Jul 23, 2010 06:12 PM)Dic? Wrote: [\[CITE\]](#) Many posts like this but, Thanks for the share.

I can see you put effort into it.

ya I've seen a few other posts but i wanted to kinda explain more in this one and i find a few people finding it useful I'll go back and explain things even simpler but thanks for your reply

sbrg - July 24, 2010, 4:33 am

Your best bet would be to delete this 'tutorial'. The only thing you accomplish by posting it is demonstrating your complete ignorance when it comes to C++.

EDIT: Oh, and P.S.: I voted "No" because helpful tutorials do not cause me to head-butt myself in my face so hard that I fracture my skull.

Phoenix-Ace - July 24, 2010, 10:06 am

thats example of yours are IT and COMSCI lessons in college!!!

I dont see any " scanf " on your tutorial!!!!

what kind of programing tools do you use?

hacku4eva - July 24, 2010, 11:13 am

(Jul 24, 2010 04:33 AM)sbrg Wrote: [\[CITE\]](#) Your best bet would be to delete this 'tutorial'. The only thing you accomplish by posting it is demonstrating your complete ignorance when it comes to C++.

EDIT: Oh, and P.S.: I voted "No" because helpful tutorials do not cause me to head-butt myself in my face so hard that I fracture my skull.lol exactly as i expected from you, that's all you do in every

forum you're in
?

EDIT: you said it shows complete ignorance right? mind sharing that part then ? mr "197" IQ xD
(Jul 24, 2010 10:06 AM)Phoenix-Ace Wrote: [\[CITE\]](#) thats example of yours are IT and COMSCI lessons in college!!!

I dont see any " scanf " on your tutorial!!!!

what kind of programing tools do you use?
ya that's how i learned it (still learning but more advanced things) in college so thought I'd share it on here

sbrg - July 24, 2010, 11:54 am

(Jul 23, 2010 05:32 PM)hacku4eva Wrote: [\[CITE\]](#) the Pre-processor headers are always your step zero in coding, you'll need to add those in order to use the built in functions that are in c++.

the Pre-processor header you'll see the most in your coding as you start learning C++ is
Code `#include <iostream>`this pre-processor header lets you access the built in function to input and output things from the command screen..like so
Code `#include <iostream>`

using namespace std; // std stands for standard, we'll be adding it under all the pre-processor headers in //every example for ease.

```
int main()
{
```

```
    cout << "Hello World";
    return 0;
```

```
}There is no such thing as a "preprocessor header." It isn't that wrong to say, but a header does not have anything to do with the preprocessor; the preprocessor includes the header. Nothing else. #include however is a preprocessing directive.
```

Using "using namespace std;" can and will cause namespace conflicts. If you are going to use it, use it within the scope of the function where you use a lot of objects from the std namespace, or simply prefix everything with "std::" that belongs to the std namespace.

Quote:when using the " cout << " operator it basically means the the console will output whatever you typed in between the parenthesis after the operator and it print it to the user sitting in front of the PC. "cout" is not an operator at all. cout is an object, or an instance of the ostream class.

It is a global object declared in the iostream header.

Code `$ cat /usr/include/c++/4.1/iostream | grep cout`

```
extern ostream cout;      ///< Linked to standard output
```

```
extern wostream wcout;    ///< Linked to standard output
```

Quote:The next Pre-processor header we'll learn about is

Code `#include <string>`

this one is fairly easy, it lets you store words "strings" and use them through out your program, here's an example..

Code `#include <iostream>`

`#include <string>`

using namespace std;

```

int main()
{
    int age; // this declares a integer called "num"
    string name;
    name = "hacku4eva"; // you could have aded this at the line above if you wish but i prefer coding
that way
    cout << "Is your name " << name << "?";
    cout << endl;
    cout << " please enter your age ";
    cin >> age;
    [cout << endl;
    cout << " Oh so your name is " << name << " and you are " << age << " years old";
    return 0;
} It's still not a "preprocesor header",

```

Quote:C) " cout << endl " that is being used from the first Pre-processor header we talked about and it simply just starts a new line. std::endl is in fact a stream manipulator that a) inserts a newline and b) flushes the output buffer.

Quote:If/else statements & switch statements

==

if/else statement are used to check statements either if they're [equivalent, not equivalent, less than/ greater than, and/or, less than or equal to or greater than or equal to] here's an example..

Now to see that same exact code using switch statements

==

Code switch (day)

```

{
    case 3:
        cout << " too late..";
        break;
    case 2:
    case 1:
        cout << " you're finally here";
        break;
    default:
        cout << "just go home";
        break;
}

```

if you looked after case 3 you'll see that right under the cout statement there is something called "break" and what that does is that it basically separates each case from the other. No. In a loop, the break statement terminates the exeuction of the innermost loop it's contained in and resumed execution right after it. Same goes for switch statements; it stops all cases from executing and resumes execution after the switch statement.

Quote:Functions

==

i'll show a value returning function using a int

Code #include <iostream>

#include <string>

using namespace std;

int test();

```
int main ()
{
    cout << "Hi" << endl;
    test();
    return 0;
}
```

```
int test()
{
    cout << " this is a test" ;
    return 0;
}
```

} if you look on top of the main function you'll see Code int test(); this is called a prototype and is used to call other functions because if you didn't include prototypes the compiler while run through the main function and it will hit the Code test(); and wont know what that is because you didn't tell it to go and get that function at first. All of this, and much more of this tutorial is incredibly badly phrased, and the layout is absolutely horrible.

On top of that, you're trying to demonstrate return values of functions, using a function that does nothing but output something, then returns a zero, when you simply ignore the return value. If anyone read this that didn't understand return values of functions isn't any wiser after reading this.

All in all, your contribution fits well into the already chock full, figurative trash can this forum is for tutorials on programming languages.

The points you got right weren't really right. Your use of proper terminology was lacking, and the layout was completely fucking retarded, grammar was horrid and the language was pathetic.

Is this enough for you?

hacku4eva - July 24, 2010, 12:00 pm

(Jul 24, 2010 11:54 AM)sbrg Wrote: [CITE](#) There is no such thing as a "preprocessor header." It isn't that wrong to say, but a header does not have anything to do with the preprocessor; the preprocessor includes the header. Nothing else.
#include however is a preprocessing directive.

Using "using namespace std;" can and will cause namespace conflicts. If you are going to use it, use it within the scope of the function where you use a lot of objects from the std namespace, or simply prefix everything with "std::" that belongs to the std namespace.

Quote:when using the " cout << " operator it basically means the the console will output whatever you typed in between the parenthesis after the operator and it print it to the user sitting in front of the PC. "cout" is not an operator at all. cout is an object, or an instance of the ostream class.

It is a global object declared in the iostream header.

```
Code $ cat /usr/include/c++/4.1/iostream | grep cout
extern ostream cout;      ///< Linked to standard output
extern wostream wcout;    ///< Linked to standard output
```

Quote:The next Pre-processor header we'll learn about is

```
Code #include <string>
```

this one is fairly easy, it lets you store words "strings" and use them through out your program, here's an example..

```
Code #include <iostream>
#include <string>
```

```
using namespace std;
```

```
int main()
{
    int age; // this declares a integer called "num"
    string name;
    name = "hacku4eva"; // you could have aded this at the line above if you wish but i prefer coding
that way
    cout << "Is your name " << name << "?";
    cout << endl;
    cout << " please enter your age ";
    cin >> age;
    [cout << endl;
    cout << " Oh so your name is " << name << " and you are " << age << " years old";
    return 0;
} It's still not a "preprocesor header",
```

std::endl is in fact a stream manipulator that a) inserts a newline and b) flushes the output buffer.

Quote:If/else statements & switch statements

==

if/else statement are used to check statements either if they're [equivalent, not equivalent, less than/ greater than, and/or, less than or equal to or greater than or equal to] here's an example..

Now to see that same exact code using switch statements

==

Code switch (day)

```
{
    case 3:
        cout << " too late..";
        break;
    case 2:
    case 1:
        cout << " you're finally here";
        break;
    default:
        cout << "just go home";
        break; if you looked after case 3 you'll see that right under the cout statement there is something
called "break" and what that does is that it basically separates each case from the other. No. In a
loop, the break statement terminates the exeuction of the innermost loop it's contained in and
resumed execution right after it. Same goes for switch statements; it stops all cases from executing
and resumes execution after the switch statement.
```

Quote:Functions

==

i'll show a value returning function using a int

Code #include <iostream>

#include <string>

```
using namespace std;
```

```
int test();
```

```
int main ()
{
    cout << "Hi" << endl;
    test();
    return 0;
}
```

```
int test()
{
    cout << " this is a test" ;
    return 0;
}
```

if you look on top of the main function you'll see Code int test(); this is called a prototype and is used to call other functions because if you didn't include prototypes the compiler while run through the main function and it will hit the Code test(); and wont know what that is because you didn't tell it to go and get that function at first. All of this, and much more of this tutorial is incredibly badly phrased, and the layout is absolutely horrible.

On top of that, you're trying to demonstrate return values of functions, using a function that does nothing but output something, then returns a zero, when you simply ignore the return value. If anyone read this that didn't understand return values of functions isn't any wiser after reading this.

All in all, your contribution fits well into the already chock full, figurative trash can this forum is for tutorials on programming languages.

The points you got right weren't really right. Your use of proper terminology was lacking, and the layout was completely fucking retarded, grammar was horrid and the language was pathetic.

Is this enough for you?

lol last i checked printing out things was part of the basics and i used the return zero to show the difference when using a value returning function and non-value one....and i said I'd explain more later on but this one was meant to just show basic things and how you'd see them.... yea i agree that the layout was retarded but that's because its my first long post but hey if you didn't like it or didn't think it was useful then fine.

the whole point of this was just to show the basics of the language

g4143 - July 24, 2010, 12:02 pm

(Jul 24, 2010 11:54 AM)sbrg Wrote: [CITE](#) There is no such thing as a "preprocessor header." It isn't that wrong to say, but a header does not have anything to do with the preprocessor; the preprocessor includes the header. Nothing else.

#include however is a preprocessing directive.

Using "using namespace std;" can and will cause namespace conflicts. If you are going to use it, use it within the scope of the function where you use a lot of objects from the std namespace, or simply prefix everything with "std:." that belongs to the std namespace.

Quote:when using the " cout << " operator it basically means the the console will output whatever you typed in between the parenthesis after the operator and it print it to the user sitting in front of the PC. "cout" is not an operator at all. cout is an object, or an instance of the ostream class.

It is a global object declared in the iostream header.

Code \$ cat /usr/include/c++/4.1/iostream | grep cout

extern ostream cout; //< Linked to standard output

extern wostream wcout; //< Linked to standard output

Quote:The next Pre-processor header we'll learn about is

Code #include <string>

this one is fairly easy, it lets you store words "strings" and use them through out your program, here's an example..

Code #include <iostream>

#include <string>

using namespace std;

int main()

{

int age; // this declares a integer called "num"

string name;

name = "hacku4eva"; // you could have aded this at the line above if you wish but i prefer coding that way

cout << "Is your name " << name << "?";

cout << endl;

cout << " please enter your age ";

cin >> age;

[cout << endl;

cout << " Oh so your name is " << name << " and you are " << age << " years old";

return 0;

} It's still not a "preprocesor header",

std::endl is in fact a stream manipulator that a) inserts a newline and b) flushes the output buffer.

Quote:If/else statements & switch statements

==

if/else statement are used to check statements either if they're [equivalent, not equivalent, less than/ greater than, and/or, less than or equal to or greater than or equal to] here's an example..

Now to see that same exact code using switch statements

==

Code switch (day)

{

case 3:

cout << " too late..";

break;

case 2:

case 1:

cout << " you're finally here";

break;

default:

cout << "just go home";

break; if you looked after case 3 you'll see that right under the cout statement there is something called "break" and what that does is that it basically separates each case from the other. No. In a loop, the break statement terminates the exeuction of the innermost loop it's contained in and resumed execution right after it. Same goes for switch statements; it stops all cases from executing and resumes execution after the switch statement.

Quote:Functions

==

i'll show a value returning function using a int

Code #include <iostream>

```
#include <string>
```

```
using namespace std;
```

```
int test();
```

```
int main ()  
{  
    cout << "Hi" << endl;  
    test();  
    return 0;  
}
```

```
int test()  
{  
    cout << " this is a test" ;  
    return 0;  
}
```

} if you look on top of the main function you'll see Code int test(); this is called a prototype and is used to call other functions because if you didn't include prototypes the compiler while run through the main function and it will hit the Code test(); and wont know what that is because you didn't tell it to go and get that function at first. All of this, and much more of this tutorial is incredibly badly phrased, and the layout is absolutely horrible.

On top of that, you're trying to demonstrate return values of functions, using a function that does nothing but output something, then returns a zero, when you simply ignore the return value. If anyone read this that didn't understand return values of functions isn't any wiser after reading this.

All in all, your contribution fits well into the already chock full, figurative trash can this forum is for tutorials on programming languages.

The points you got right weren't really right. Your use of proper terminology was lacking, and the layout was completely fucking retarded, grammar was horrid and the language was pathetic.

Is this enough for you?

The worst thing....He asked for it. Some people never learn....

"EDIT: you said it shows complete ignorance right? mind sharing that part then ? mr "197" IQ xD"

hacku4eva - July 24, 2010, 12:08 pm

(Jul 24, 2010 12:02 PM)g4143 Wrote: [\[CITE\]](#) "cout" is not an operator at all. cout is an object, or an instance of the ostream class.

It is a global object declared in the iostream header.

Code \$ cat /usr/include/c++/4.1/iostream | grep cout

```
extern ostream cout;    ///< Linked to standard output
```

```
extern wostream wcout;  ///< Linked to standard output
```

Quote:The next Pre-processor header we'll learn about is

```
Code #include <string>
```

this one is fairly easy, it lets you store words "strings" and use them through out your program, here's an example..

```
Code #include <iostream>
```

```
#include <string>
```

```
using namespace std;
```

```

int main()
{
    int age;    // this declares a integer called "num"
    string name;
    name = "hacku4eva"; // you could have added this at the line above if you wish but i prefer coding
that way
    cout << "Is your name " << name << "?";
    cout << endl;
    cout << " please enter your age ";
    cin >> age;
    [cout << endl;
    cout << " Oh so your name is " << name << " and you are " << age << " years old";
    return 0;
} It's still not a "preprocessor header",

```

std::endl is in fact a stream manipulator that a) inserts a newline and b) flushes the output buffer.

Quote:If/else statements & switch statements

==

if/else statement are used to check statements either if they're [equivalent, not equivalent, less than/ greater than, and/or, less than or equal to or greater than or equal to] here's an example..

Now to see that same exact code using switch statements

==

Code switch (day)

```

{
    case 3:
        cout << " too late..";
        break;
    case 2:
    case 1:
        cout << " you're finally here";
        break;
    default:
        cout << "just go home";
        break;
}

```

*if you looked after case 3 you'll see that right under the cout statement there is something called "break" and what that does is that it basically separates each case from the other. No. In a loop, the break statement terminates the execution of the innermost loop it's contained in and resumes execution right after it. Same goes for switch statements; it stops all cases from executing and resumes execution after the switch statement.

All of this, and much more of this tutorial is incredibly badly phrased, and the layout is absolutely horrible.

On top of that, you're trying to demonstrate return values of functions, using a function that does nothing but output something, then returns a zero, when you simply ignore the return value. If anyone read this that didn't understand return values of functions isn't any wiser after reading this.

All in all, your contribution fits well into the already chock full, figurative trash can this forum is for tutorials on programming languages.

The points you got right weren't really right. Your use of proper terminology was lacking, and the layout was completely fucking retarded, grammar was horrid and the language was pathetic.

Is this enough for you?

The worst thing....He asked for it. Some people never learn....

"EDIT: you said it shows complete ignorance right? mind sharing that part then ? mr "197" IQ xD"
[/quote]

i was just sharing it as i learned it this way and i thought it was easy but ok

sbrg - July 24, 2010, 12:15 pm

(Jul 24, 2010 12:08 PM)hacku4eva Wrote: [\[CITE\]](#) It's still not a "preprocesor header",

std::endl is in fact a stream manipulator that a) inserts a newline and b) flushes the output buffer.

Quote:If/else statements & switch statements

==

if/else statement are used to check statements either if they're [equivalent, not equivalent, less than/ greater than, and/or, less than or equal to or greater than or equal to] here's an example..

Now to see that same exact code using switch statements

==

Code switch (day)

```
{
    case 3:
        cout << " too late..";
        break;
    case 2:
    case 1:
        cout << " you're finally here";
        break;
    default:
        cout << "just go home";
```

break;*if you looked after case 3 you'll see that right under the cout statement there is something called "break" and what that does is that it basically separates each case from the other. No. In a loop, the break statement terminates the exeuction of the innermost loop it's contained in and resumed execution right after it. Same goes for switch statements; it stops all cases from executing and resumes execution after the switch statement.

All of this, and much more of this tutorial is incredibly badly phrased, and the layout is absolutely horrible.

On top of that, you're trying to demonstrate return values of functions, using a function that does nothing but output something, then returns a zero, when you simply ignore the return value. If anyone read this that didn't understand return values of functions isn't any wiser after reading this.

All in all, your contribution fits well into the already chock full, figurative trash can this forum is for tutorials on programming languages.

The points you got right weren't really right. Your use of proper terminology was lacking, and the layout was completely fucking retarded, grammar was horrid and the language was pathetic.

Is this enough for you?

The worst thing....He asked for it. Some people never learn....

"EDIT: you said it shows complete ignorance right? mind sharing that part then ? mr "197" IQ xD"
[/quote]

i was just sharing it as i learned it this way and i thought it was easy but ok
[/quote]

Look man, there is nothing wrong with the desire to share one's knowledge. But if you really want to be good at something, you should always realize that you know nothing, and always be sure that what you know is entirely correct.

If you want to share information, it should not be misinformation. A few googles along the way of your tutorial, some better layout and it would have been fine.

But for now, it is probably best for you to keep learning on your own. Ask questions, write a shitton of code, and then when you really are in a place where sharing your knowledge makes sense, do so, if you are confident you can share correct information and so so properly.

hacku4eva - July 24, 2010, 12:18 pm

(Jul 24, 2010 12:15 PM)sbrg Wrote: [\[CITE\]](#) No. In a loop, the break statement terminates the execution of the innermost loop it's contained in and resumed execution right after it. Same goes for switch statements; it stops all cases from executing and resumes execution after the switch statement.

All of this, and much more of this tutorial is incredibly badly phrased, and the layout is absolutely horrible.

On top of that, you're trying to demonstrate return values of functions, using a function that does nothing but output something, then returns a zero, when you simply ignore the return value. If anyone read this that didn't understand return values of functions isn't any wiser after reading this.

All in all, your contribution fits well into the already chock full, figurative trash can this forum is for tutorials on programming languages.

The points you got right weren't really right. Your use of proper terminology was lacking, and the layout was completely fucking retarded, grammar was horrid and the language was pathetic.

Is this enough for you?

The worst thing....He asked for it. Some people never learn....

"EDIT: you said it shows complete ignorance right? mind sharing that part then ? mr "197" IQ xD"
[/quote]

i was just sharing it as i learned it this way and i thought it was easy but ok
[/quote]

Look man, there is nothing wrong with the desire to share one's knowledge. But if you really want to be good at something, you should always realize that you know nothing, and always be sure that what you know is entirely correct.

If you want to share information, it should not be misinformation. A few googles along the way of your tutorial, some better layout and it would have been fine.

But for now, it is probably best for you to keep learning on your own. Ask questions, write a shitton of code, and then when you really are in a place where sharing your knowledge makes sense, do so, if you are confident you can share correct information and so so properly.
[/quote]

i never said i was all pro i was simply sharing what i've learned so far this semester...ok besides the bad layout and the lack of explaining things clear, did my code have any sort of mistakes that i need to work on ? (not trying to sound cocky) but so i can know and fix it

sbrg - July 24, 2010, 12:21 pm

(Jul 24, 2010 12:18 PM)hacku4eva Wrote: [\[CITE\]](#) i never said i was all pro i was simply sharing what i've learned so far this semester...ok besides the bad layout and the lack of explaining things clear, did my code have any sort of mistakes that i need to work on ? (not trying to sound cocky) but so i can know and fix it

No, you are right. You did not say that you were all pro. But the last thing people need when they are even newer than you are people that are rather new themselves, potentially misleading them with information they think is correct.

But no, your code was alright, though it seems like you are using tabs instead of spaces, which fucks up the indentation when you post it here. Or you are simply using bad indentation.

hacku4eva - July 24, 2010, 12:25 pm

(Jul 24, 2010 12:21 PM)sbrg Wrote: [\[CITE\]](#) No, you are right. You did not say that you were all pro. But the last thing people need when they are even newer than you are people that are rather new themselves, potentially misleading them with information they think is correct.

But no, your code was alright, though it seems like you are using tabs instead of spaces, which fucks up the indentation when you post it here. Or you are simply using bad indentation.

i tried to use the tab on here but wasn't really working so i kept spacing them but forgot for some parts i guess..and yea you got a point like i mean i simply understood it that way and was able to catch up quickly but not everyone grasps information the same way

Phoenix-Ace - July 24, 2010, 6:11 pm

then try to make a tutorial ion JAVA!!!

or

can you make program virus in c++?

LOL!!!!

hahaha

hacku4eva - July 24, 2010, 9:35 pm

(Jul 24, 2010 06:11 PM)Phoenix-Ace Wrote: [\[CITE\]](#) then try to make a tutorial ion JAVA!!!

or

can you make program virus in c++?

LOL!!!!

hahaha

if i knew java i'd be hanging there..

and nope, not yet at least but once i do i'll be sure to share it