

*Visual Studio.net/Dev C++
Tutorial*

TABLE OF CONTENTS

Introduction	3
Dev C++	3
Acquiring the Software	3
Launching Dev C++.....	3
Composing a Program.....	4
Compiling Your Program	6
Executing Your Program	7
Compiling in Dev C++ with multiple files	7
Debugging Your Programs	10
<i>Compilation Errors or Why won't this thing compile?</i>	10
<i>Runtime Errors or It compiles but the average of 3 and 2 does not equal to 0!</i>	11
Microsoft Visual Studio.NET	12
Acquiring the Software	12
Launching Microsoft Visual Studio.NET	12
Composing a Program.....	13
Compiling and Executing the Program.....	14
Using the Debugger	16
Generating Output.....	18
<i>Print the Screen</i>	18
<i>Capture the Contents of the Command-line Console Screen</i>	18
Evaluation.....	19

Introduction

This tutorial is not designed to teach a programming language or how to program in C++ but is intended to provide an introduction to and give assistance with two software development tools, DEV C++ and Visual Studio.net both available to students in Texas State Computer Science Department computer labs. This tutorial assumes the participant has at least a minimum amount of programming knowledge (having written at least three programs in any programming language).

Dev C++

Dev-C++ by Bloodshed Software is a good, simple, free IDE (Integrated Development Environment) for the C/C++ programming language. It provides a GUI (Graphical User Interface) on top of the popular GCC (GNU Compiler Collection). It is often favored over Microsoft's Visual Studio or Borland's C++ Builder because of its simplicity. For small programs such as those used to teach beginning level programming classes it's a very good development environment choice. For larger and more complex programs, commercial products offer more features.

Acquiring the Software

Dev-C++ is available via download from the following locations.

- <ftp://ftp.cs.txstate.edu/pub/cs/os/windows/devC/>
- www.bloodshed.net

It is now also included on the Computer Science "Getting Started" CD that is available for a \$3 handling/production cost fee.

Launching Dev C++

It's time to open up Dev C++ and take a look at what it's all about. To launch Dev C++ from most of the computers in the Computer Science Department, click on **Start Menu** → **All Programs** → **Development software** → **Dev C++** → **Dev-C++** (see Figure 1.0).

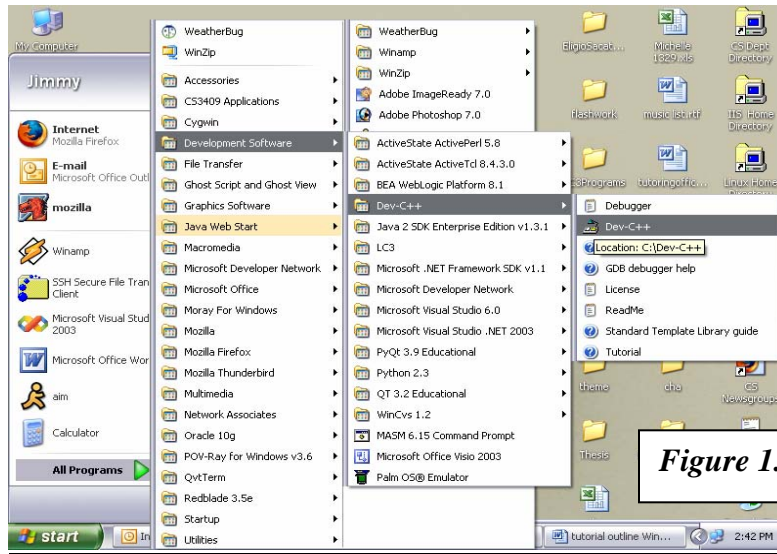


Figure 1.0

The opening screen of Dev C++ is displayed below. (see Figure 1.1)

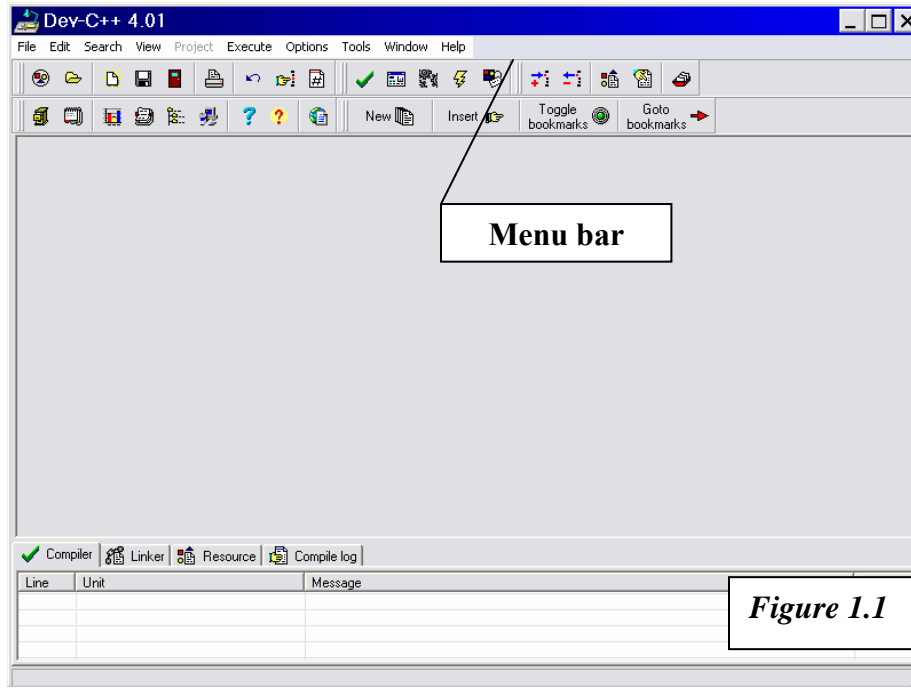
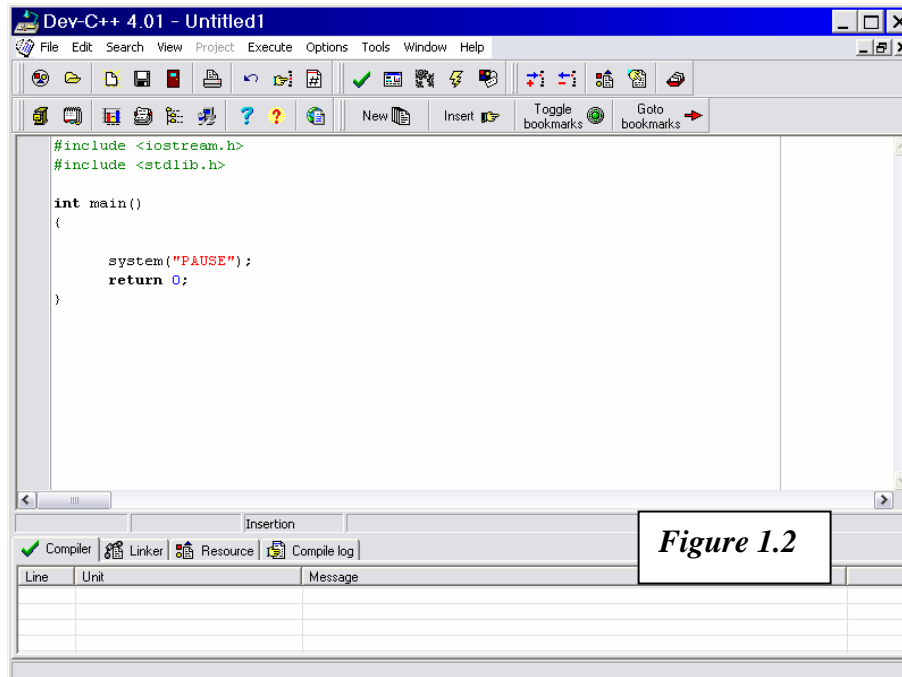


Figure 1.1

Composing a Program

To compose a simple program in Dev C++, first click on the 'File' on the top menu bar and choose the option 'new source file'. *You should do this now and continue to follow along as we go through some examples.*

Dev C++ automatically generates a source file and provides a starting basic format. (see Figure 1.2)



Example/Exercise 1

Our first example program receives an input of a person's name and then outputs it to a console screen with an attached message. The program code is displayed below and looks like Figure 1.4 when types in.

```
#include <iostream.h>
#include <stdlib.h>

int main()
{
    char name[10];

    cout<<"input a name:"<<endl;
    cin>> name;
    cout<<"Hello "<<name<<" welcome to the C++ programming tutorial
        section"<<endl;

    system("PAUSE");
    return 0;
}
```

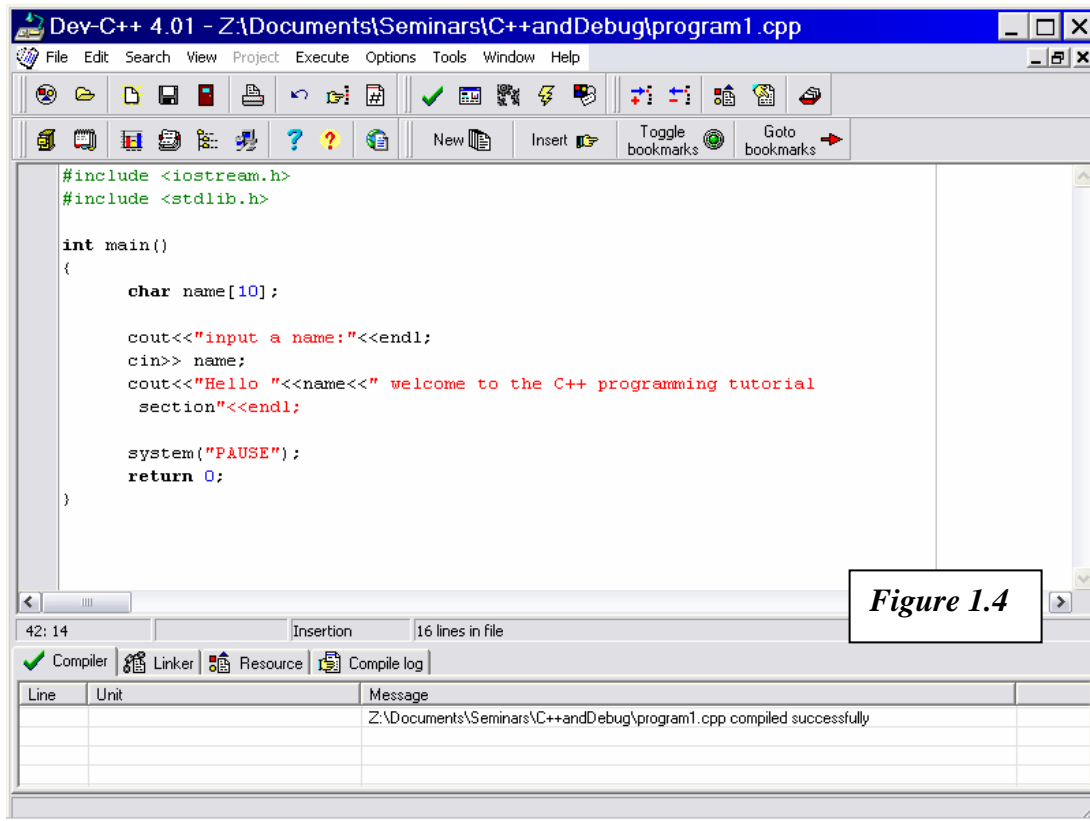


Figure 1.4

After you are done typing in the code remember to save. Select File and save the unit as *program0.cpp*

Compiling Your Program

To compile your source code, click ‘**execute**’ on the menu bar and click on ‘**compile**’. A window will appear in the middle and indicate whether or not there are any errors in your program. (see Figure 1.5)

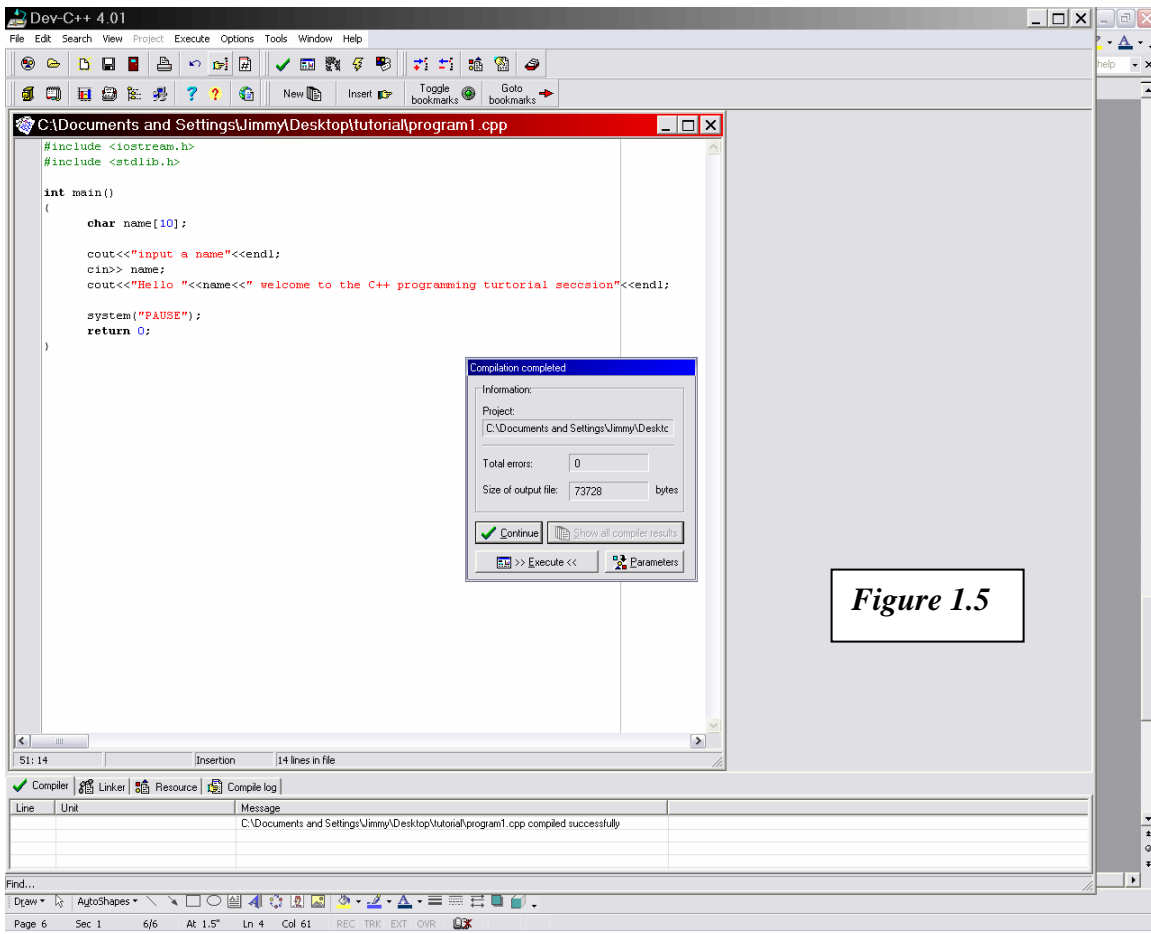


Figure 1.5

Executing Your Program

To execute the compiled program, click on '>>Execute<<'. A black console screen appears, and you will see a message to **“input a name:”**. Type in a name and hit ENTER. A message similar to that below should appear.

**“Hello “name” welcome to the C++ programming tutorial section
Press any key to continue . . .”**

You have just written, compiled and executed a program in the Dev C++ environment!

Compiling in Dev C++ with multiple files

More often than not a C/C++ program is written using multiples files. Let’s re-write the same program in a manner which will require the compilation of three different source files: a header file, an implementation file and an application file. (figure 1.7)
For this example we will need to create a project.

Click on the menu bar and click on **“new project”**. This brings up a window and prompts for a name for the project. Call it project1.

```
#ifndef PROGRAMH_H
#define PROGRAMH_H
#include <iostream.h>
#include <stdlib.h>

void print_funtion(char name[10]);

#endif
```

On the menu bar, click **'file'** then **"New source file"** and type in the text displayed in the box at the right.

On the menu bar, click **'file'** and save this unit as **"program1h.h"**

Repeat the process for the next source code file. On the menu bar, click **'File'** then **"New source file"**. Type in the code displayed in the following box.

```
#include "program1h.h"

void print_funtion(char name[10])
{
    cout<<"Welcome! "<<name<<" , to our C++ tutorial
        section."<<endl;
}
```

On the menu bar select **'File'** and save this unit as **"program1i.cpp"**

Repeat once more for the third file. On the menu bar click **'File'** then **"New source file"** and type in the following text.

```
#include "program1h.h"

int main()
{
    char name[10];
    cout<<"Input your name:"<<endl;
    cin>>name;
    print_funtion(name);
    system("PAUSE");
    return 0;
}
```

On the menu bar select **'File'** and save this unit as **"program1a.cpp"**.

When you have finished the screen on your computer should resemble the figure on the next page. Watch out for typographical mistakes!

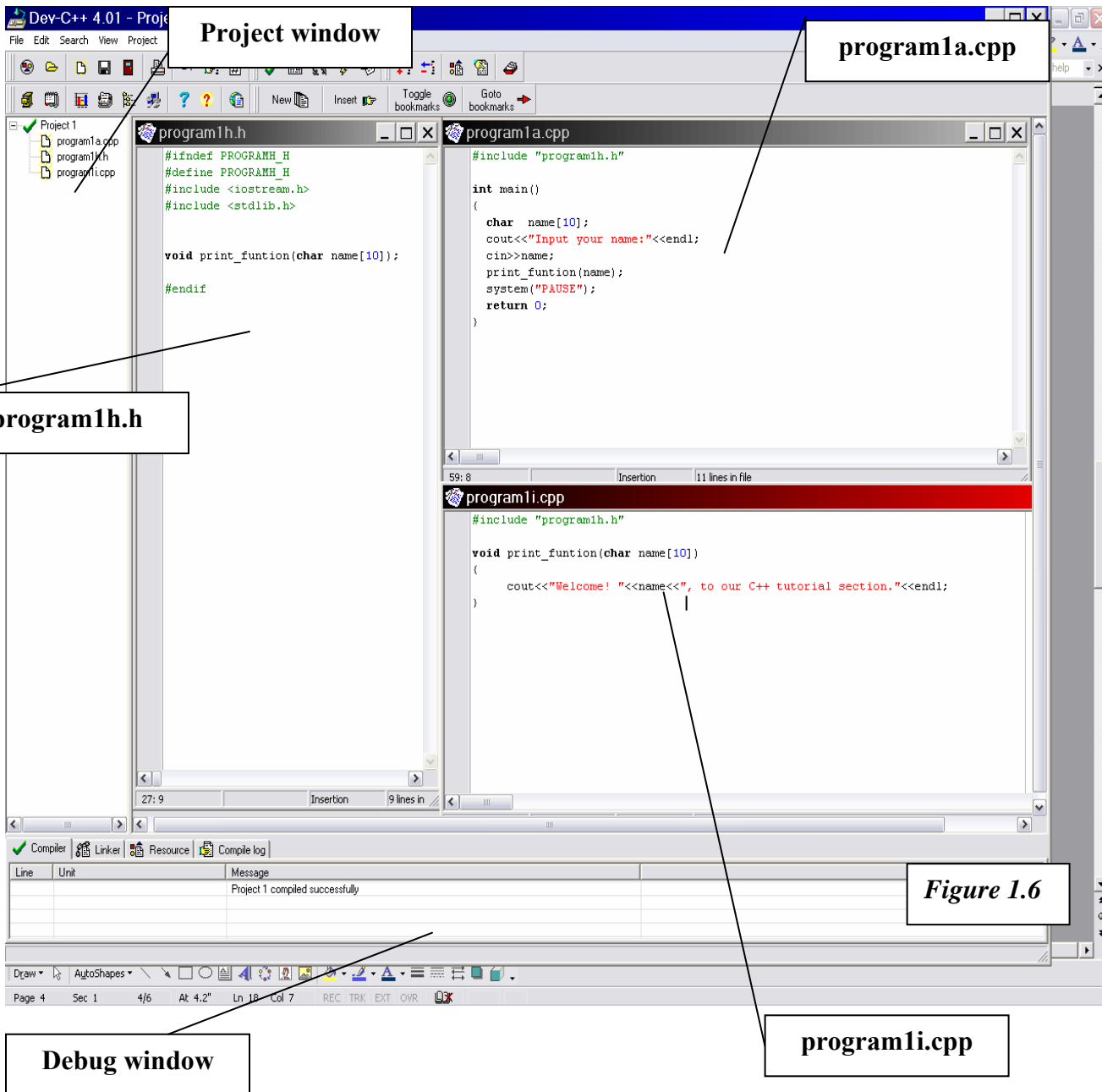


Figure 1.6

On the menu bar select **'File'** and **execute** and **"compile"** and you should get the same result as *program0*. Okay we are done with this program now go to **'File'** and click **'close project'**.

Debugging Your Programs

Compilation Errors or Why won't this thing compile?

Everyone make mistakes, even the experts. Sometimes finding these mistakes is easy but sometimes a little help would be welcome. Dev C++ provides some assistance in this regard with respect to compilation errors. The following program takes an input of two numbers and outputs the result to the console.

Create a “**new source file**” and type in the following program code.

```
#include <iostream.h>
#include <stdlib.h>

int main()
{
    int count;
    float a,b,average;
    cout<<"enter a number"<<endl
    cin>>a;
    cout<<"enter a 2nd number"endl;
    cin>>b;
    sum=a+b;
    average=sum/count;
    cout<<"The sum of the two numbers is: "<<sum<<endl;
    cout<<"The average is: "<<average<<endl;
    system("PAUSE");
    return 0;
}
```

Save the file as program2.cpp.

On the menu bar select ‘**File**’ and **execute** and “**compile**”. There are errors and it doesn’t compile. So what happened?

Look at the debug window down at the bottom of the window. (see figure 1.7)

- Double click on the 1st line. This takes you to ‘line 9’ in the program and the error message says “parse error before ‘>’”. This message is indicating an error before this line. If you take a look at ‘line 8’, you may notice a missing semi-colon at the end of the line.
- Click on the 2nd line on the debug window. This time line 10 is highlighted and the error message says “parse error before ;”. On line 10, “<<” was left out the before the endl.
- Click on the 3rd line of the debug window. This message indicates that the variable ‘sum’ is not declared. Go to line 7 where you see **float average;** add one more variable declaration on a new line below: **float average,sum;** . This should correct the last problem.

Compile again by selecting ‘**File**’ and **execute** and “**compile**” and TADA! It works this time.

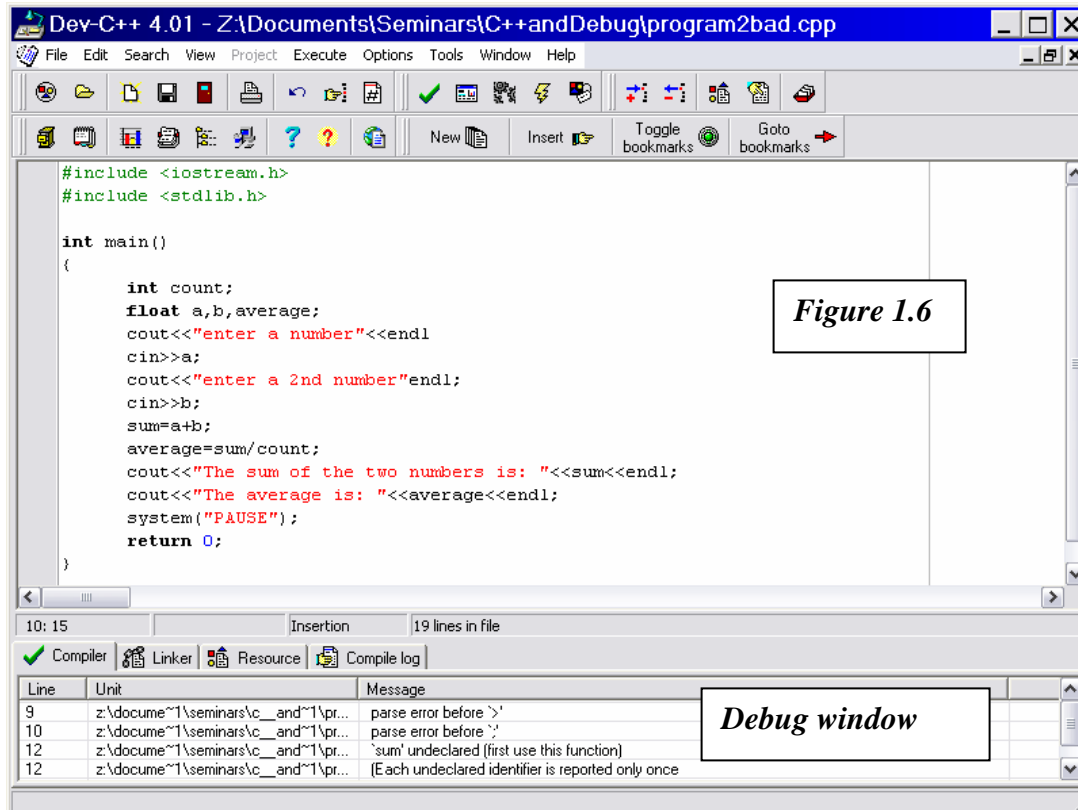


Figure 1.6

Debug window

Input two numbers as requested in the console screen. So why do we have a weird number for our average???

Runtime Errors or It compiles but the average of 3 and 2 does not equal to 0!

Print (cout) statements are your best friend when trying to debug complicated programs. Well placed print statements allow you to keep track of the value of variables at each step of the execution.

In our example we are pretty sure the sum value is correct. That leaves the count value, since the formula for average is $average = sum / count$. Go ahead and print out the value for count.

Add the line `cout<<"The count is "<<count<<endl;` right after `sum=a+b;` and compile it. We see the value for count is not 2 so set the value of the count equal to 2, change the line 6 to `int a,b,count=2;` and compile it again

Whew...we got everything right this time! ☺

Microsoft Visual Studio.NET

With free development software like Dev-C++ around, why do we bother with commercial software like Visual Studio.NET? With Visual Studio.NET developers can deliver a range of professional software in record time. The integrated development environment (IDE) provides a consistent interface for all languages, including Microsoft Visual Basic .NET, Microsoft Visual C++ .NET, Microsoft Visual C# .NET and Microsoft Visual J# .NET. So, to make it short, it's a lot more powerful than Dev C++.

Acquiring the Software

As long as you are currently registered for a degree related Computer Science course, Visual Studio.NET can be obtained via your MSDNAA (Microsoft Developer's Network Academic Alliance) account. These accounts are active for all eligible students from the 12th class day of a semester through the day grades are due at the Registrar's office. Your username will be your Texas State email address (ex. xx1234@txstate.edu). If you do not know your password you can have it emailed to your Texas State email at the login screen. Go to <http://auster.cs.txstate.edu/> to access the Texas State Computer Science Department MSDNAA site.

Launching Microsoft Visual Studio.NET

Microsoft Visual Studio.NET can be accessed on most of the computers in the Computer Science Department by clicking on **Start Menu** → **All Programs** → **Development software** → **Microsoft Visual Studio.NET** → **Microsoft Visual Studio.NET** (see figure 2.0).

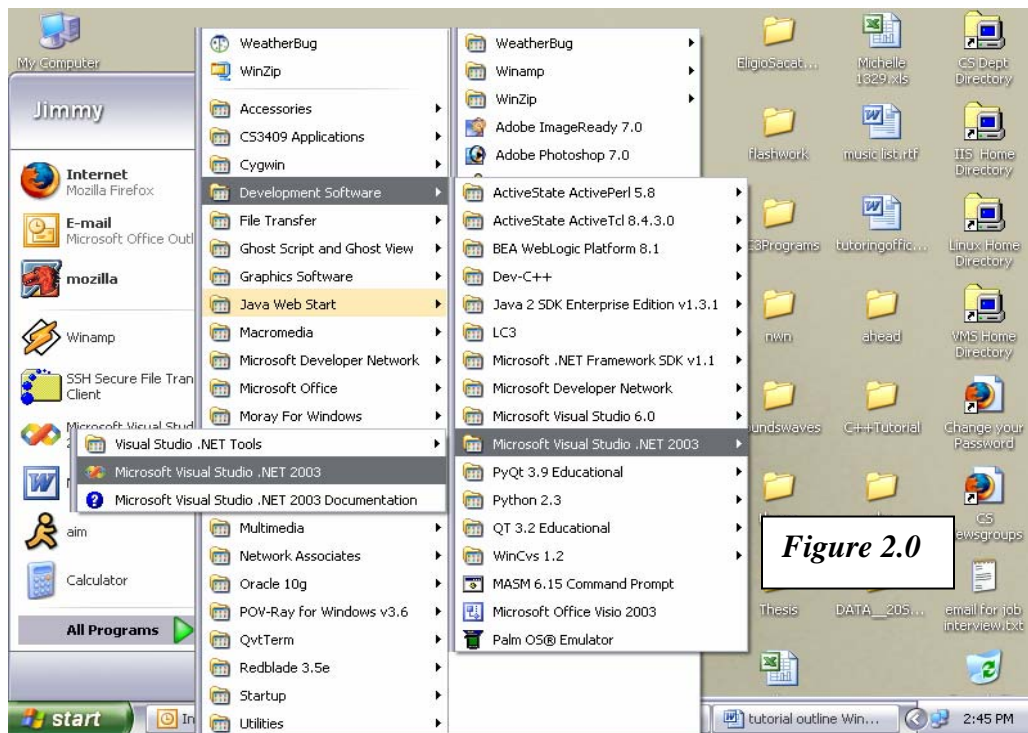
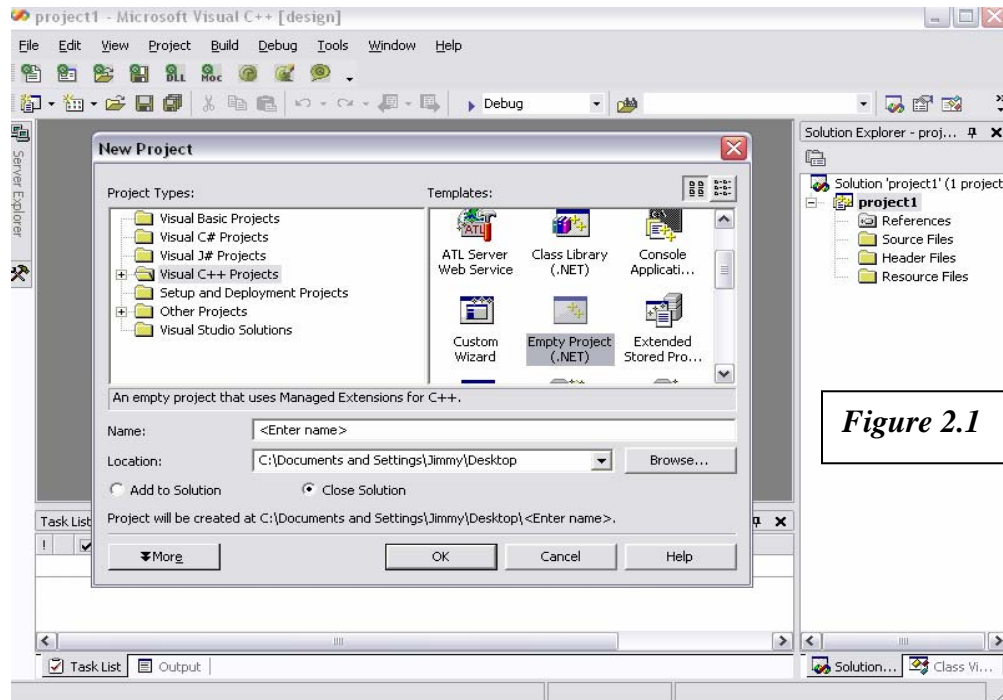


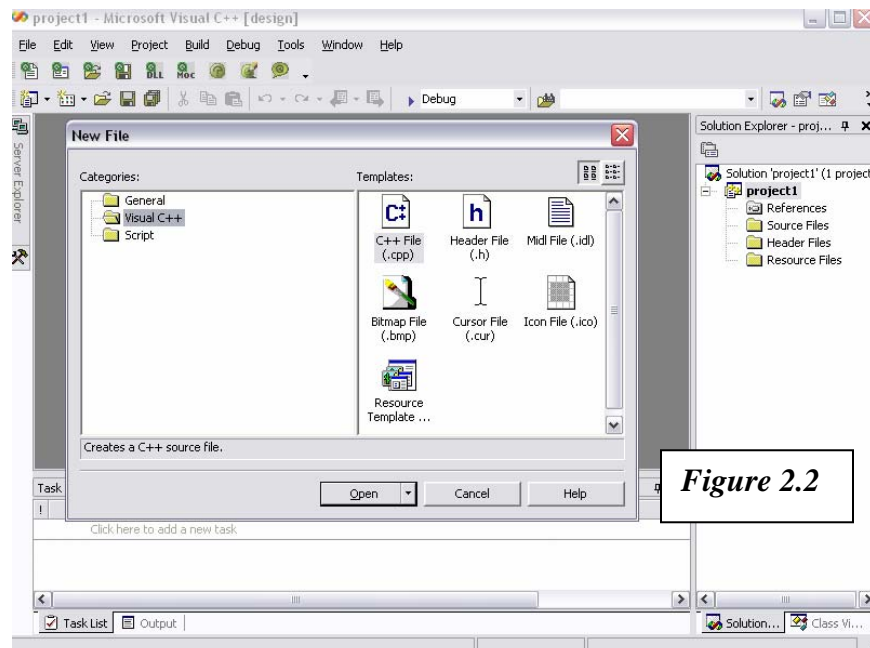
Figure 2.0

Composing a Program

On the menu bar, select **File → New → Project**. There are many options. Select the options, **Visual C++ Projects** and **Manage C++ Empty Project**. Enter a name and location for saving for the project (Ex. C:\Documents and Settings\Bob\Desktop\project1). (see Figure 2.1) *Note: Please execute these commands now and follow along as we go through the tutorial.*



A project which can consist of one or more files has been created. Now we need to create a file in which to compose code. Click on **File → New → File**. In the window (see Figure 2.2) select the folder **Visual C++** and choose the **C++ file (.cpp)**. A text editor will open up a window and you can start entering source code.



This time, however, to save a little time, instead of typing in our source code directly, we will port source code into the development environment by copying and pasting text from an existing file. Using Windows Explorer, open a file called **program3.txt** from the **C++Tutorial** folder on the Floppy/CD provided. Copy and paste the code into the cpp file within Visual Studio.NET. The program code is displayed in the text box below.

The program sums up all the integers from 0 to an input value. It contains several looping structures (for and while).

```
//This program is sums up all the integers starting from 0 to the input integer
#include<iostream>
#include<stdlib.h>
using namespace std;

int main()
{
    //declare and initialize variables
    int input,sum=0
    char choice='y';

    while(choice=='y' || choice=='Y')    //test whether user wants to run the
                                        //program again
    {
        input=-1;
        while(input<0) //while loop to test for positive integer input
        {
            cout<<"Please input a positive integer: ";
            cin>>input;
        }
        for(i=0; i<input; i++) //for loop to sum all
                               //positive integers less than the input
        {
            sum=sum+i;
        }
        cout<<"The sum from 0 to integer "<<input<<" is: "<<sum<<endl<<endl;
        cout<<"Do you want to run it again? ";
        cin>>choice;

        //reinitzation
        input=-1;
        sum=0;
    }

    return(0);
}
```

Compiling and Executing the Program

Go ahead and save the file as program3.cpp and make sure the cpp file is inserted into the project. You can do that by right click on the cpp file and select the last option “**move program3.cpp into project1**”

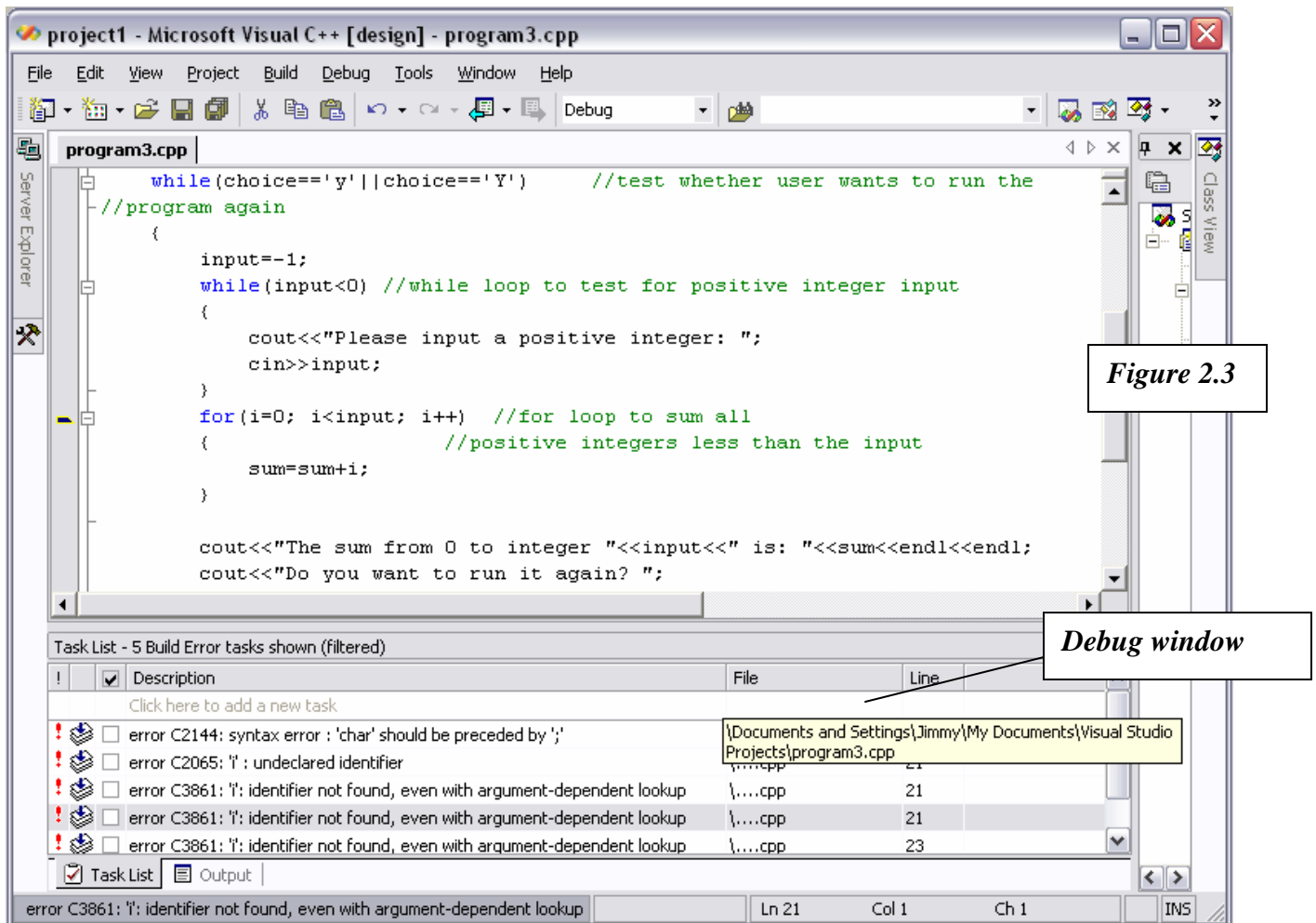
Click **Debug** on the **File** menu and click on **Start without debugging**. A window will prompt for whether or not to **Build**. Click **Yes**. Another window should pop up and say “There were build errors. Continue?”. Click **No**. We will now have to figure out why the program does not compile. ☹

Look at the debugging window down at the bottom. (see Figure 2.3)

- The first line says “*syntax error: ‘char should be preceded by ;’*” at line 10. Double clicking on the error takes you to line 10. However, the problem is actually on the preceding line 9 which is missing a semicolon.
- The second line says “*‘i’ : undeclared identifier*”. Double clicking on that error takes you to line 21. The variable *i* is not declared. To declare it change line 21 to the following:

```
for(int i=0; i<=input; i++)
```

Recompile the code, **Debug -> Start without Debugging** and this time the program compiles.



When the console window prompts for a positive integer, type in the number “4”. The expected result is 10 since $0+1+2+3+4=10$. Hmm..., the display shows 6. The program compiles but does not produce correct results.

Using the Debugger

As mentioned previously, one method of locating runtime problems is to strategically place ‘print’ statements throughout the code to trace the value of variables as the code executes. Although this process is a little time consuming, it is highly recommended for beginning programmers as a great learning tool. However, Visual Studio.NET provides an alternative troubleshooting procedure in the form of a sophisticated, powerful Debugger tool. So when the program compiles and runs without syntax error and the result is not what as expected, there may be a logical error or errors involved, using the Debugger in Visual Studio.NET can help us find the problem. Below are a few of the features of the Debugger that are quite useful.

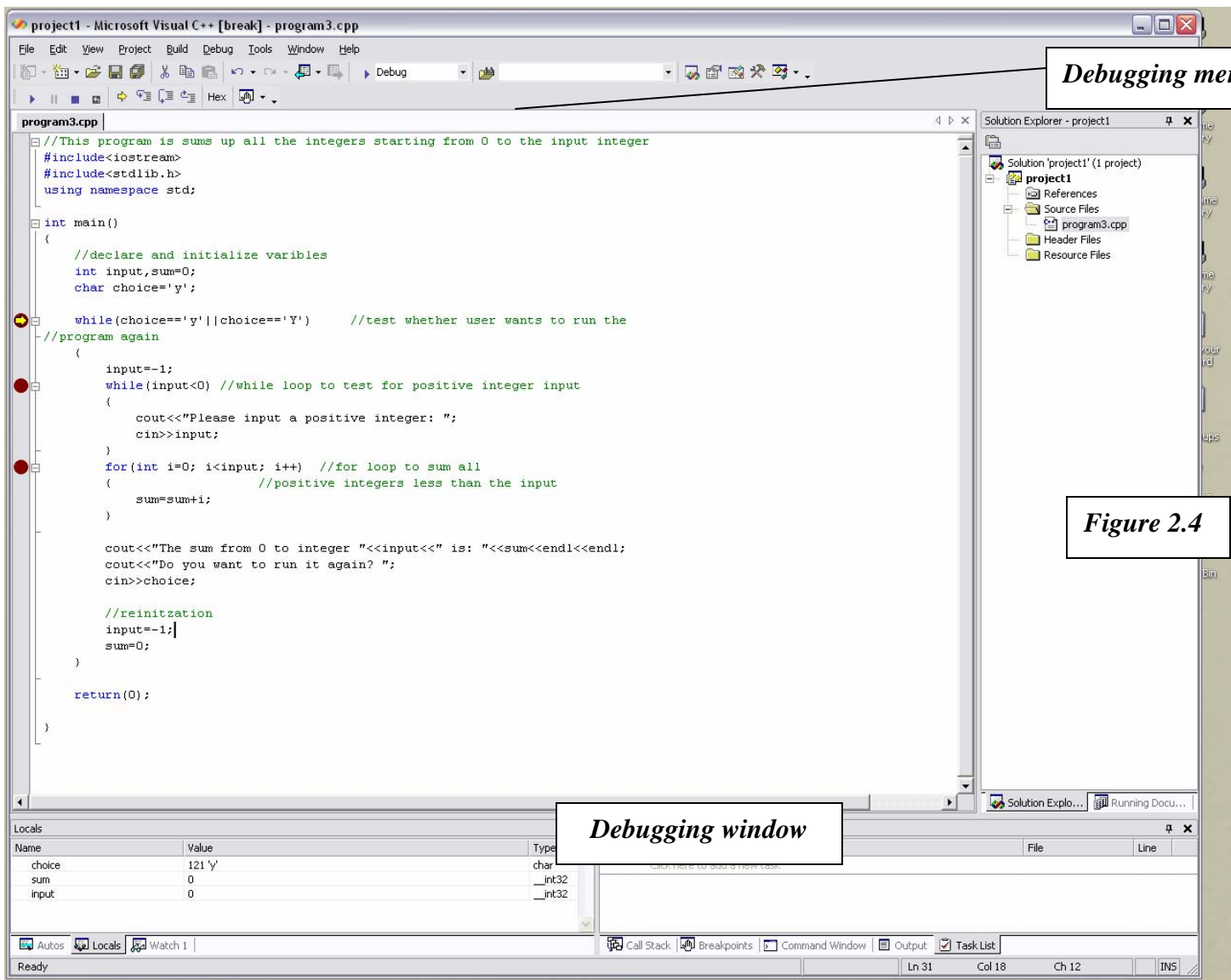
- **Break points** – A break point is a marker that can be inserted at a specified location. Break points allow the programmer to keep track of the value of variables at a particular instance.
- **Step Into** – Step into a function call.
- **Step Over** – Step over a sequence of statements, loops and so on.
- **Step Out** – Step out of a function call while one is in it.

Also there is a little handy feature that one can jump to a specific line without browsing through the code. Click on **edit -> Go To** and type in the line number you wish to jump to.

Now we will use some of these features to troubleshoot the misbehaving program.

First we will set some break points at lines 12, 15, and 21. Go to each line and, on the very left of the window, click on it once. A red circle should appear. Now compile the program this time using **Debug → Start** and click **Yes**. The bottom left window (see Figure 2.4), displays the values (below) of several variables at that point of execution in the program.

choice	121	'y'	char
sum	0	__	int32
input	0	__	int32



Debugging menu

Figure 2.4

Debugging window

You should also see a new menu bar pop up from the top displaying icons for **Step Into**, **Step Over** and **Step Out** (see Figure 2.4). Click on **Step Over**. A yellow arrow at the left side will move to the next line. Click it again and it moves to line 15. We can see that the value of the variable named input changes to -1. Click on the **Step Over** button two more times. The console window now prompts the user to input a positive integer. Go ahead and input our luck number 4 again. Once a number has been entered, the input variable value from the debugging window now changes to that value.

Keep clicking on **Step Over**, a new variable i is now initialized to 0. Continue to **Step Over** and watch variable i start to increment by 1 each time and the variable sum value change at each addition. However, when it stops at the entered integer, it is not added to sum. This is a common place for logic errors to occur, at the boundaries (beginnings and ends) of looping structures. To correct the error, add an equal sign in front of the input within the for loop statement as show below:

```
for(int i=0; i<=input; i++)
```

Now recompile and execute the program again. This time it should produce the expected result (0+1+2+3+4=**10**) 😊 !

Generating Output

Most program assignments require that an output be submitted along with the program. If an output file is generated then that file is submitted. But how do you submit output that is sent to the console screen? There is more than one way to generate a copy of this type of output but the method used may depend on which way the professor wants it done. Usually one of the following two methods will suffice.

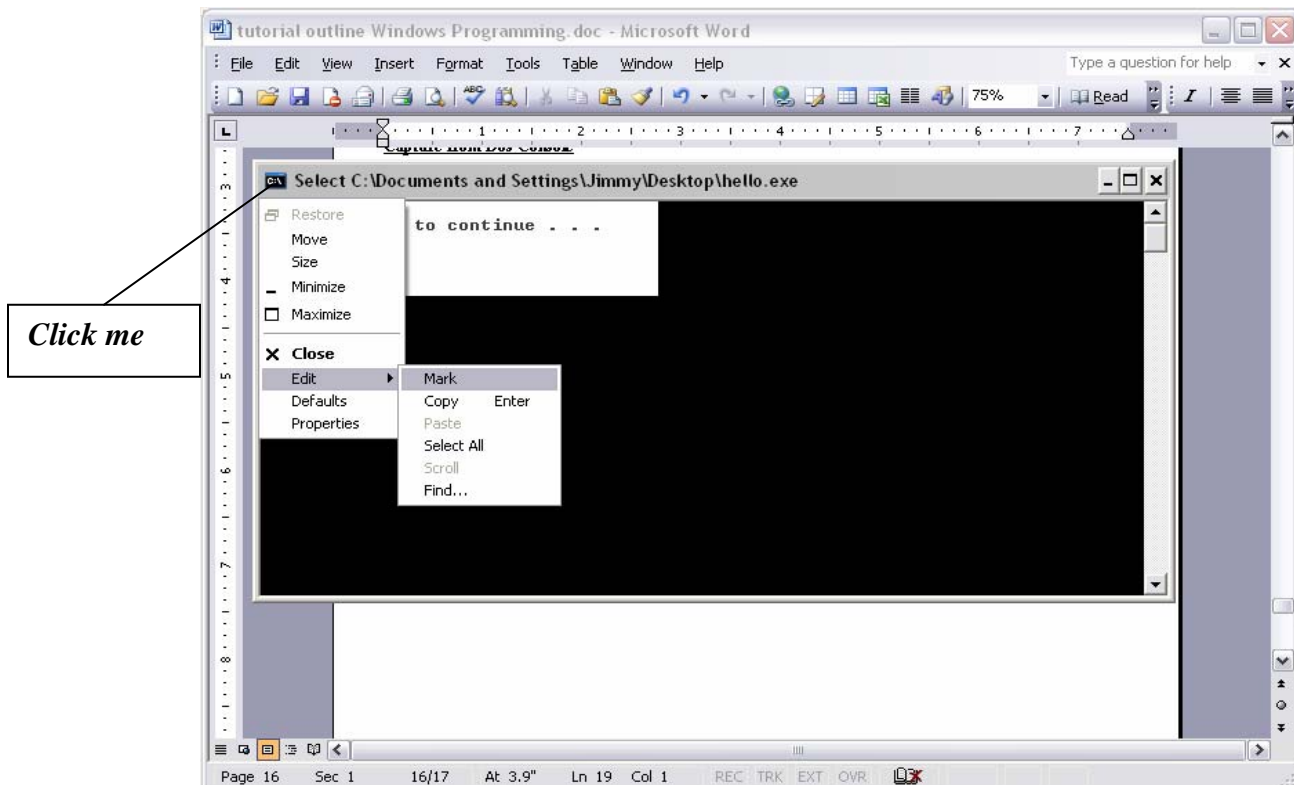
Print the Screen

The easiest method is to print a copy of the entire screen. Once you have generated the output result just hit the “Print Screen Key” to capture the entire current screen to the paste buffer. You can then call up WordPad, Microsoft Word or even Paint and do a paste operation to get a printable version of the screen shot which includes your output.

If you want to capture only the window containing your output, make certain that that particular window is the active window on your screen and hit <ALT> <PrtScn> to put a copy of only the active screen into the paste buffer. You will still have to paste the contents of the buffer into some other program such as a word processor in order to print it.

Capture the Contents of the Command-line Console Screen

Generate your output to in the command-line console. Click on the little symbol at the left hand corner of the console and select **Edit -> Mark**. Then highlight the desired text and hit enter (this will put the highlighted text into the paste buffer. This can now be pasted into another program as before. However since this will just be text, Notepad can also be used to generate the file to print.



Evaluation

The intention of this tutorial is to familiarize the participant with the development environments provided by Dev C++ and Visual Studio.NET. Often this valuable and time saving information is not provided by instructors whose main concern is the teaching of the skills and techniques required to write good code. We hope that the tutorial is a good complement to your programming course.

Please leave your comments regarding the tutorial and indicate any areas improvement you feel should to be made.

Finally, thanks for coming and we hope this tutorial has proved useful and beneficial.

Comments:
