Part III

Appendices

Appendix A

Technical Instructions

Learning Outcomes

This material will help you learn how to use the software you need to do your work in this course. You won't be tested on it.

Learning Activities

- Install the Python software, if you're working with your own computer.
- Follow along with the Python instructions yourself and make sure you can work with the tools.
- Explore the software more on your own.

Topic A.1 Installing Python

We are going to use Python to write and run Python programs in this course. The following tutorial will help you get familiar with some of the functionality of the Python software.

This installation tutorial assumes that you're using Windows. Python is available for the MacOS and for Linux as well. You can use any operating system for your work in this course. You can also use Python in a computer lab on-campus. If you do, Python is already installed and you can skip to the next topic.

You can download the most recent version of Python from the Python web site, http://www.python.org/download/. Click on the link that says: "Python

2.*x*.*x* Windows installer" (where 2.*x*.*x* is the most recent release of version 2). Save this file on your desktop.

Do not download Python 3: it contains some incompatibilities that this Guide (and most other Python tutorials) do not take into account. For the a Macintosh, download the 32-bit version of Python, not the 64-bit version.

Once the file has downloaded, double-click the installation file. You can safely accept all of the defaults for the installation. Make sure you install at least the "Python interpreter and libraries" and "Tck/Tk". You will need those for this course. You will probably want the help files as well.

You can delete the installation file you downloaded once the installation is complete.

Topic A.2 Using Python

There are two different interfaces where you can write Python code: *IDLE* (Integrated DeveLopment Environment) or the Command Line. We will use IDLE in this course since it provides a graphical interface for you to work with. You can start IDLE by selecting "IDLE (Python GUI)" from the Start menu if you're using Windows.

Note that in this sections, the screen shots are from Windows, but the instructions apply to any operating system.

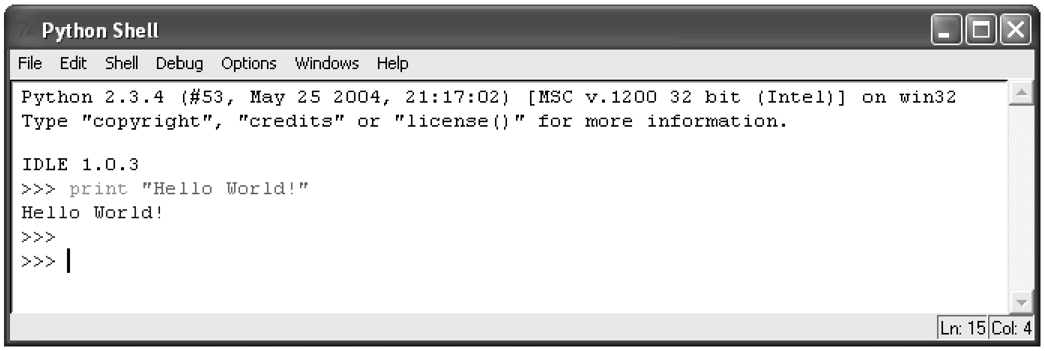
The usual first program that's written in every programming language is one that prints "Hello World" on the screen. Let's see how we can keep up the old tradition in Python. Open up the IDLE if you haven't already. Your IDLE window will have some text similar to this:

```
Python 2.3.4 (#53, May 25 2004, 21:17:02) [MSC v.1200 32
bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more
information.

IDLE 1.0.3
>>>
```

The `>>>` at the bottom is the prompt for writing the statements. When you open up IDLE, your cursor should by default be in front of this prompt. Type in the following statement in IDLE:

```
print "Hello World!"
```



```
Python 2.3.4 (#53, May 25 2004, 21:17:02) [MSC v.1200 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.

IDLE 1.0.3
>>> print "Hello World!"
Hello World!
>>>
>>> 
```

Figure A.1: IDLE, after running a "Hello World" statement

You have just executed your first Python statement and your IDLE window should look like Figure A.1.

Your first Python program

Typing statements in the IDLE environment isn't the only way to execute Python statements. You can make programs, store them and run them later. These programs are also called script files and are usually saved with a .py extension. Let's make a simple script and run it. Select "New Window" from the File menu in IDLE. An editor window will appear on your screen.

Type the same statement as you did in the IDLE earlier:

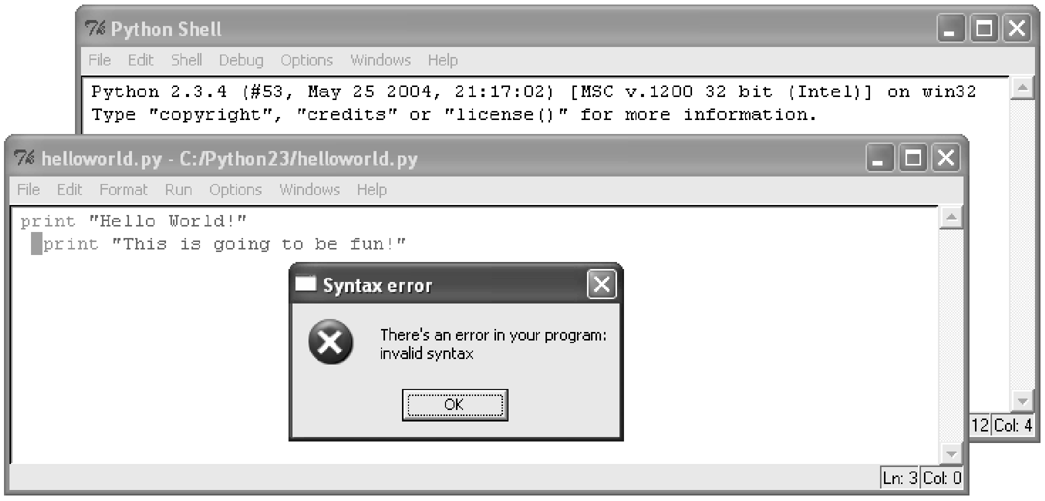
```
print "Hello World!"
```

Select the "Save As…" option from the File menu and save the program as HelloWorld.py. Don't forget the .py extension.

Now run this script file. The easiest way to run and debug your script file is to press F5 while your script file's editor window. The script file should run in the main IDLE window, displaying its output below the other output that previous commands have created.

If you change your script file and try to run it, you will be asked if you want to save your file—you must save the program before it can be run. Change the "Hello World" program that you just made: add the following print statement after the first one:

```
print "This is going to be fun!"
```



```
Python 2.3.4 (#53, May 25 2004, 21:17:02) [MSC v.1200 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
```

```
print "Hello World!"
 print "This is going to be fun!"
```

Syntax error

There's an error in your program:
invalid syntax

Python Shell
File Edit Shell Debug Options Windows Help
helloworld.py - C:/Python23/helloworld.py
File Edit Format Run Options Windows Help
OK
Ln: 3 Col: 0
12 Col: 4

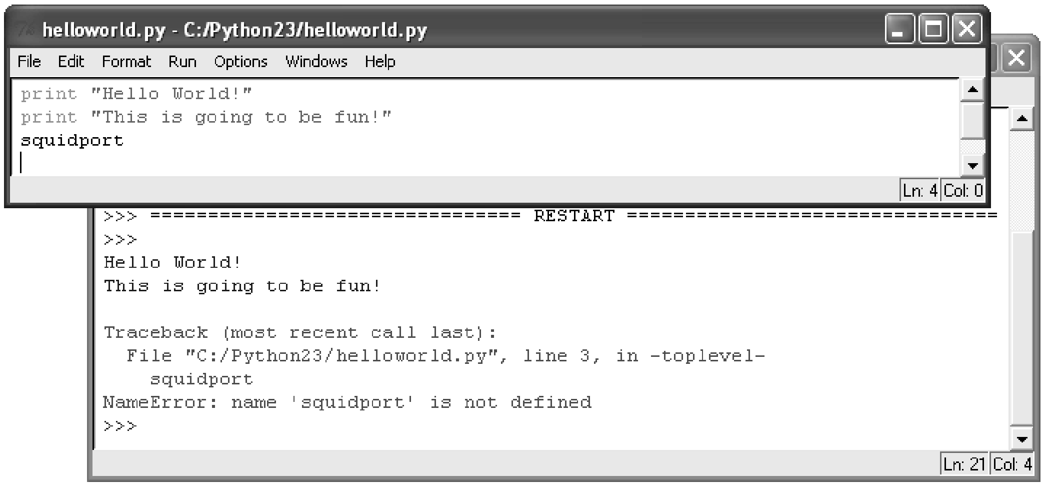
Figure A.2: IDLE, displaying a syntax error

Press F5 now. The following window will appear asking you to save the source first. Click on OK and the IDLE will display the out put of your script file.

If there are any syntax errors in your script file, those are identified automatically before the file is run. For example, in Figure A.2 an error message popped up when F5 was pressed. The error here is the incorrect indenting of the second `print` statement—the cursor is moved the the interpreter's best guess at the location of the error. Indentation plays a vital role in Python programming. You will learn more about this as you proceed in the course.

Once any syntax errors are fixed, the script will run. There might be more errors that the interpreter can't catch until its running the program. For example, in Figure A.3, the interpreter has caught an error. When it got to the word `squidport`, it didn't know what to do with it. It can't catch this any earlier since you might have created a variable or function called `squidport`; the only way to tell for the computer to tell for sure was to run it and see.

The error message indicates that it noticed the error on line 3 of the current program ("`-toplevel-`"). The IDLE editor tells you the line number that the cursor's on in the bottom left corner. In Figure A.3, "Ln: 4" indicates that the cursor is on line 4, just below the error. The "Ln: 21" in the interpreter window isn't what we're interested in: the error message


helloworld.py - C:/Python23/helloworld.py
File Edit Format Run Options Windows Help
Ln: 4 Col: 0
Ln: 21 Col: 4

```
print "Hello World!"
print "This is going to be fun!"
squidport
```

```
>>> ============================== RESTART ==============================
>>>
Hello World!
This is going to be fun!

Traceback (most recent call last):
  File "C:/Python23/helloworld.py", line 3, in -toplevel-
    squidport
NameError: name 'squidport' is not defined
>>>
```

Figure A.3: IDLE, displaying a run-time error

always gives lines in the source file.

Remember: if you want to save the program so you can run it later (or hand it in), it has to go in an editor window, not at the IDLE `>>>` prompt.

Topic A.3 Computing Accounts

There are several username/password combinations you need for this course. Hopefully, this will help you keep them straight. All of the accounts have the same user names, but different passwords.

- **SFU Computing Account:** This account is the one that all SFU students (and faculty and staff) get. It is used to retrieve your @sfu.ca email. All email sent to the course email list will go to this address (unless you have forwarded it elsewhere). This account is also used for Caucus, WebCT, and many other computing resources on-campus.

  You activate this account, go to my.sfu.ca and click the "Apply for ID" link. You need to enter your student number and some other personal information. You should contact the ACS Help Desk for problems with this account.

- **Gradebook Account:** This account is used to access Gradebook (http://gradebook.cs.sfu.ca). Gradebook will be used to check your marks on assignments and exams. Your Gradebook password is also needed to access some parts of the course web site. Gradebook is generally activated in the second week of the semester.

  This account is also used for the submission server, which is used to submit your work on assignments.

  Your initial password in Gradebook will be your student number (with no spaces or punctuation). If you have used Gradebook before, you password will be the same. If you have problems with your password, the instructor and TAs can reset it to your student number.

- **ACS Lab Account:** This account is used to access the computers in any of the ACS labs on campus. You can do your work for this course in these labs if you wish.

  Your password on this account is the same as your SFU Computing Account. If you have a new account, you should be able to log in with no problem. If your account is older, you will have to synchronize your Active Directory password. This is a one time process and can be done on the web. You can contact the ACS Help Desk for problems with this account.

  More information on using the labs is available from the course web site.

- **CSIL Lab Account:** This account is used to access the computers in the CSIL labs, which are run by the School of Computing Science. You can do your work for this course in these labs if you wish.

  Your password on this account is the same as your SFU Computing Account. You will need an access card to get in the door; you can get the access card by the second week of classes at the Security office.

Topic A.4 Working in the Lab

All of the software you need in this course is installed in both the ACS labs and the CSIL lab. You can access both and can work in either (or on your own computer is you prefer).

See the course web site for more information about using the labs.

Summary

This material will help you learn how to use the software you need to do your work in this course. You won't be tested on it.

If there are any updates to this material, they will be posted on the course web site or sent by email.

Index