**Error Handling**

Suppose your program asks the user for a number:

print() # just for space

age = int(input("Enter your age: "))

This code works fine. But what happens if the user does not enter a number. What if you enter a word, such as “hello”?

Enter a number: hello

Traceback (most recent call last):

File "None", line 2, in <module>

builtins.ValueError: invalid literal for int() with base 10: 'hello'

Your program breaks and an error message displays. In the real world, computer programs should never crash. Instead, your program needs to anticipate possible errors and deal with them accordingly. This is called error handling. An anticipated error is called an exception.

To catch an exception, we put the line (or lines) of code that might cause an error, and we put it in a try section. Then we put an error message in an except section. Both sections must be indented, like so:

print()

try: # try this...

age = int(input("Enter your age: "))

except: # if we found an error...

print ("Error! Invalid entry.")

**Value Error Exception vs Undesired Input**

If a program is asking for a user’s age and they enter a negative number, that is undesirable, but not illegal. An undesired response will not trigger an exception, so you can’t use try… except. For example, this will not work:

print()

try: # try this...

age = int(input("Enter your age: "))

except: # if user enters a negative

print ("Error! Age must be a positive number.")

In the above example, if you enter a negative number, it will be accepted and no error message will display. A negative number is acceptable.

If you want to check for a negative number, you must use a regular IF statement:

print()

age = int(input("Enter your age: "))

if age < 0: # if age is negative

print ("Error! Must be a positive number.")