

# Python for Beginners

Part 2

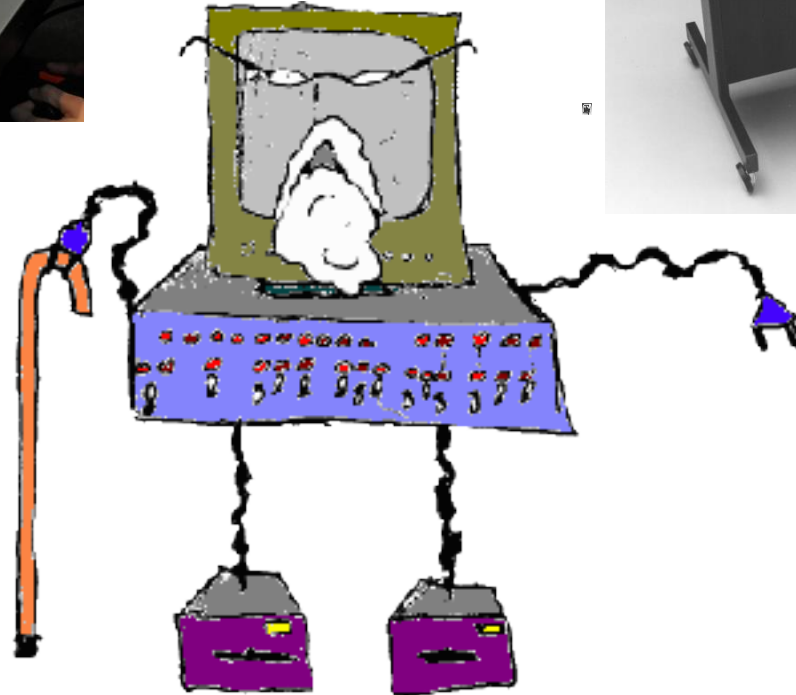
# What We Learned in Part 1

- \* Python is a great beginner language.
- \* Idle - Integrated DeveLopment Environment
- \* Idle has two modes:
  1. Interactive Mode :
    - >>> on the screen
    - responds immediately to commands
  2. Script Mode :
    - Use *File->New File* to write a program
- \* We learned Two functions :
  - \* `print()` : prints out a string
  - \* `input()` : asks for input and waits for user to type the Enter key
- \* We learned how to add comments
  - \* `# this is a human readable comment that Python ignores`

# What We Are Learning in Part 2

- \* Escape Sequences or How to print the unprintable!
- \* Triple Quoted Strings
- \* Fun With ASCII Art
- \* Variables
  - Strings
  - Integers
  - Floating Point Numbers
- \* Printing & Formatting Numbers
- \* Your Favorite Subject - MATH!

# Printing In The Good Ol' Days



# Basic Escape Sequences

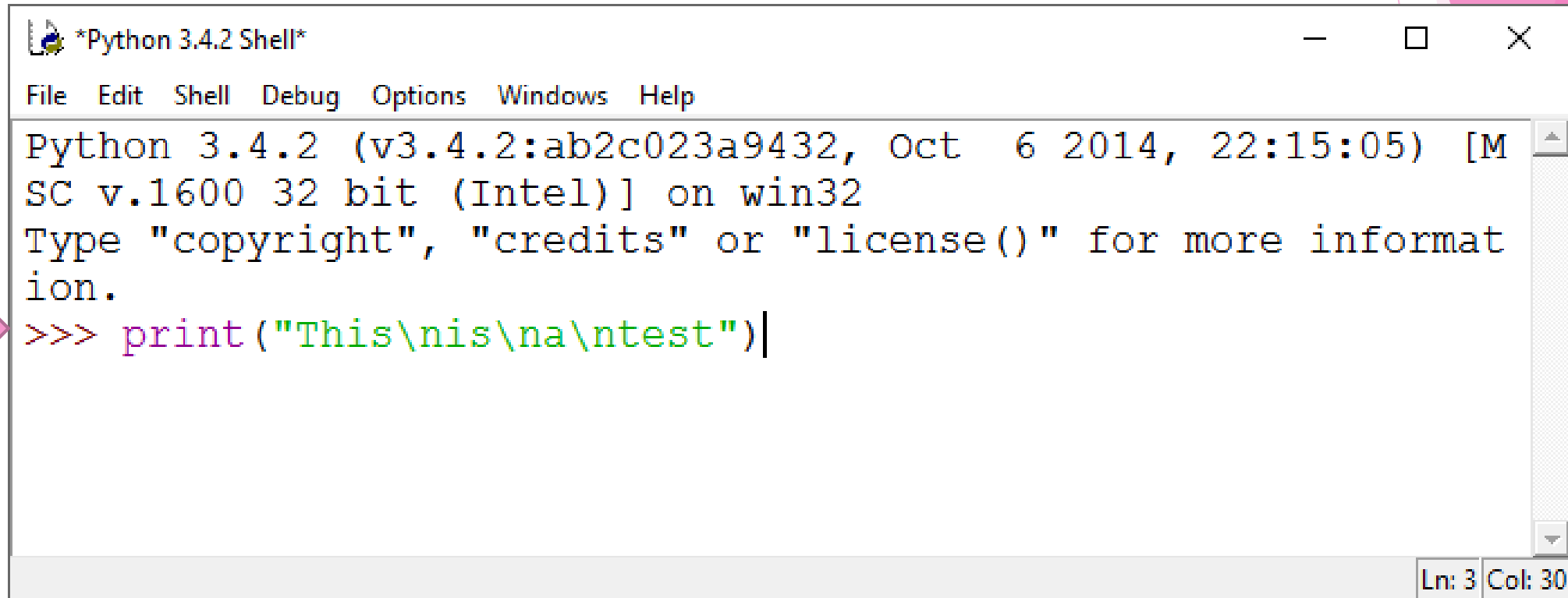
**\n** Prints on the next line.

**\t** Prints a tab.

**\a** Makes a bell sound.  
(depends on the device)

# Let's Try This in the Python Shell

Instead of spaces use \n.

A screenshot of a Python 3.4.2 Shell window. The window has a title bar with a Python logo and the text '\*Python 3.4.2 Shell\*'. Below the title bar is a menu bar with 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Windows', and 'Help'. The main text area shows the Python version and build information: 'Python 3.4.2 (v3.4.2:ab2c023a9432, Oct 6 2014, 22:15:05) [MSC v.1600 32 bit (Intel)] on win32'. It also displays the prompt 'Type "copyright", "credits" or "license()" for more information.' and the command '>>> print("This\nis\na\ntest")|'. A pink arrow points to the command line. The status bar at the bottom right shows 'Ln: 3 Col: 30'.

```
*Python 3.4.2 Shell*  
File Edit Shell Debug Options Windows Help  
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct 6 2014, 22:15:05) [MSC v.1600 32 bit (Intel)] on win32  
Type "copyright", "credits" or "license()" for more information.  
>>> print("This\nis\na\ntest")|  
Ln: 3 Col: 30
```


# Escape Sequences \n

```
>>> print("This\nis\na\ntest")
```

```
This  
is  
a  
test
```

# Let's Try This in the Python Shell

Instead of spaces use \t.

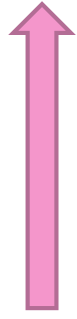
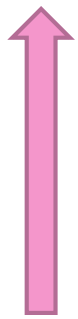
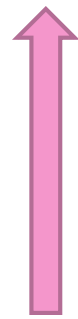


```
*Python 3.4.2 Shell*
File Edit Shell Debug Options Windows Help
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct 6 2014, 22:15:05) [MSC v
.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> print("this\tis\ta\ttest")|
Ln: 3 Col: 30
```



# Escape Sequences \t

```
>>> print("This\tis\ta\ttest")  
This      is      a      test
```



**tabs**


# Escape Sequences \a

Type these statements in a script file & save.  
Run from Windows folder by double clicking.

```
*ring.py - C:/Users/Daun/Desktop/Teaching/python/week 2/ring.py (3.4.2)*  
File Edit Format Run Options Windows Help  
print("Ringing a bell \a")  
input("\n\nPress Enter to Exit.")  
|
```

# If we use double quotes to indicate a string, then how do we print a quote?

Try This in the Shell... What Happens?




```
Python 3.4.2 Shell
File Edit Shell Debug Options Windows Help
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct 6 2014, 22:15:05) [MSC v.1600
32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> print("She said "Hello" to me.")|
```

Ln: 3 Col: 36

# Printing a Quote \"


Try it this way instead!



```
*Python 3.4.2 Shell*
File Edit Shell Debug Options Windows Help
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct 6 2014, 22:15:05) [MSC v.1600 32 bit (In
tel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> print("She said \"Hello!\" to me.")|
```

Ln: 3 Col: 39

# Printing a Quote \"

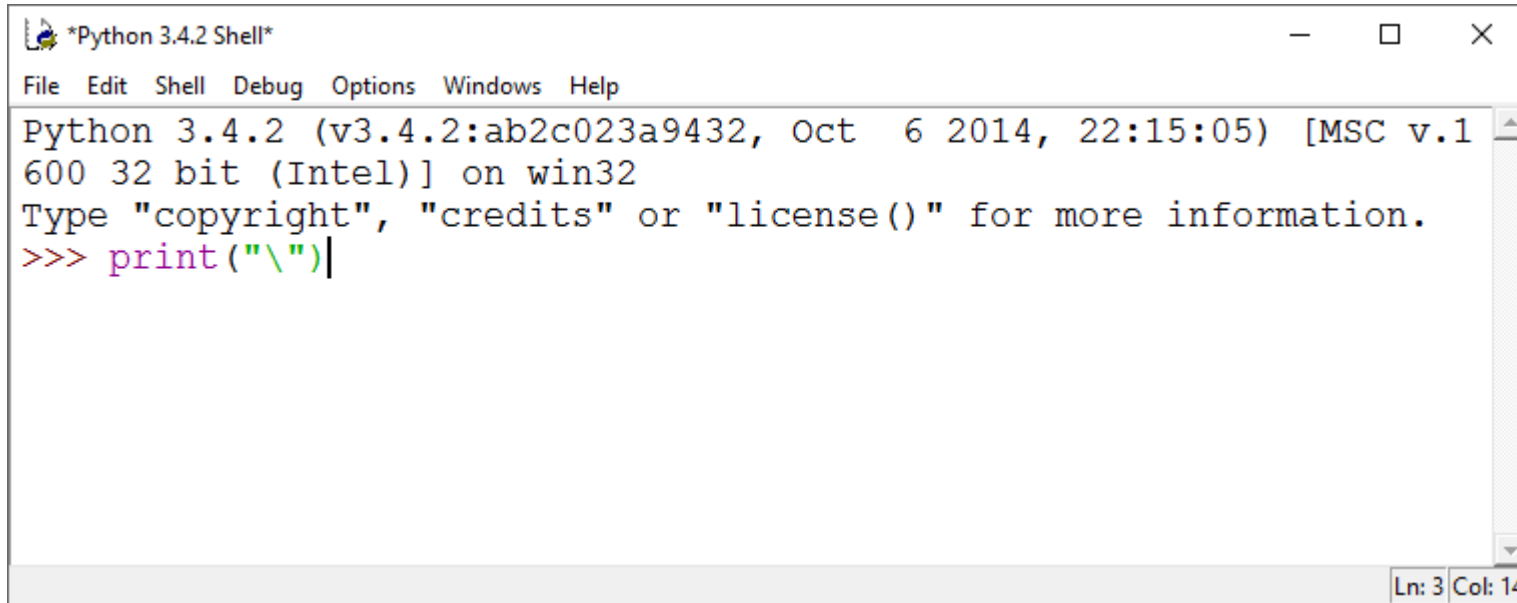


```
Python 3.4.2 Shell
File Edit Shell Debug Options Windows Help
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct 6 2014, 22:15:05) [MSC v.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> print("She said \"Hello!\" to me.")
She said "Hello!" to me.
>>> |
```

Ln: 5 Col: 4

# If we use back slashes to indicate an escape sequence, then how do we print a backslash?

Type a print statement with only one(1) backslash in the string argument.  
What happens?




```
*Python 3.4.2 Shell*  
File Edit Shell Debug Options Windows Help  
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct 6 2014, 22:15:05) [MSC v.1  
600 32 bit (Intel)] on win32  
Type "copyright", "credits" or "license()" for more information.  
>>> print("\\")|
```

The screenshot shows a Python 3.4.2 Shell window. The title bar is "\*Python 3.4.2 Shell\*". The menu bar includes File, Edit, Shell, Debug, Options, Windows, and Help. The main text area displays the Python version and build information, followed by a prompt for more information. The command `>>> print("\\")` has been entered, and a cursor is positioned at the end of the line. The status bar at the bottom right indicates "Ln: 3 Col: 14".

# Printing a Backslash \\

Type a print statement with 6 backslashes in the string argument.




```
*Python 3.4.2 Shell*
File Edit Shell Debug Options Windows Help
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct 6 2014, 22:15:05) [MSC v
.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> print("\\\\\\"|
```

Ln: 3 Col: 19

How many backslashes actually print?

# Printing a Backslash \\



```
Python 3.4.2 Shell
File Edit Shell Debug Options Windows Help
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct 6 2014, 22:15:05) [MSC v
.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> print("\\\\")
\\
>>> |
```

Ln: 5 Col: 4

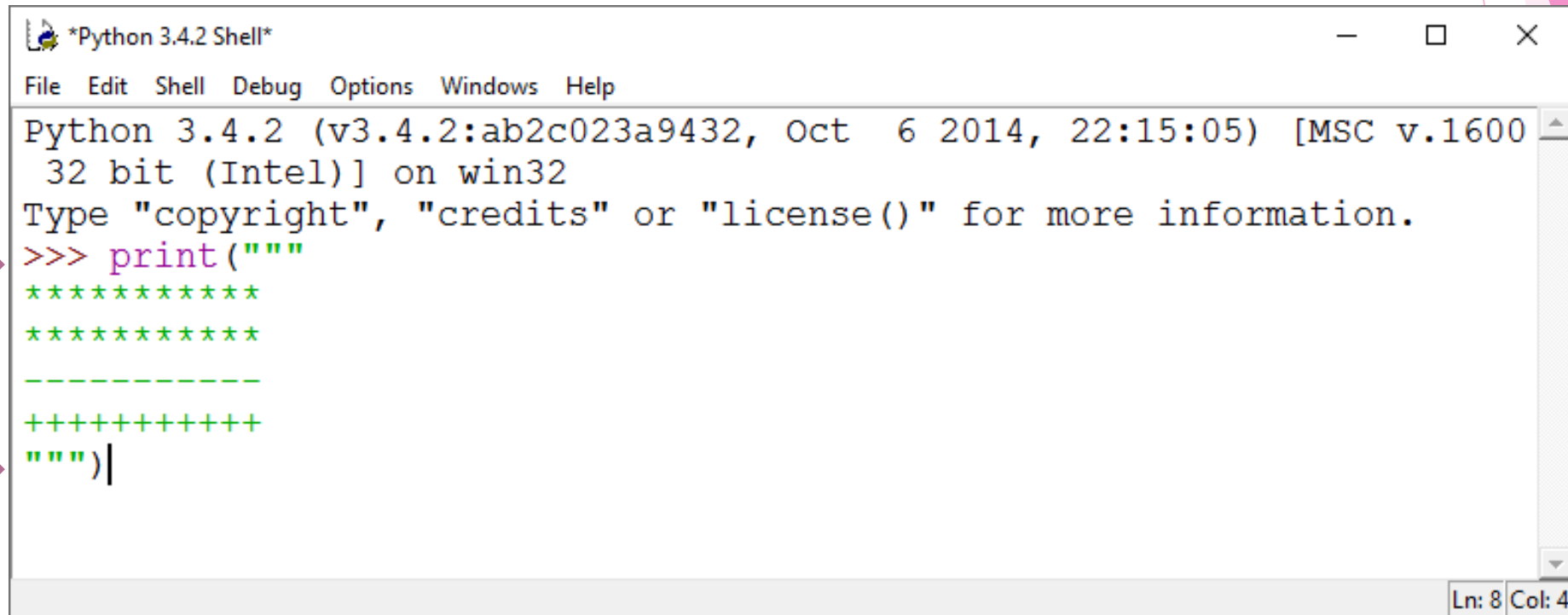


# Triple Quoted Strings

Type this ...

There are 3 quotes at the beginning and end.

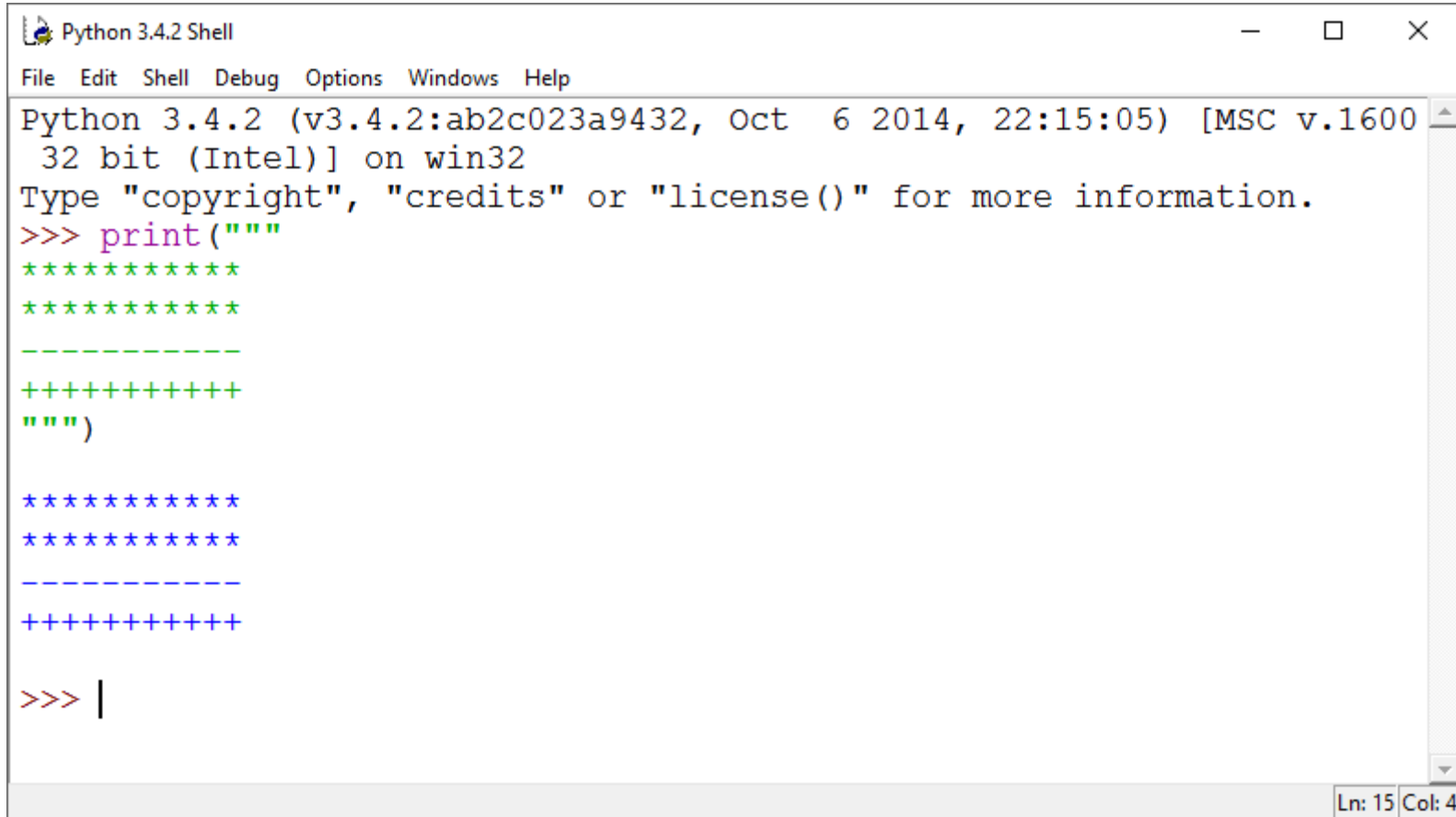
There are 11 of each character.



```
*Python 3.4.2 Shell*
File Edit Shell Debug Options Windows Help
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct 6 2014, 22:15:05) [MSC v.1600
 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> print("""
*****
*****
-----
+++++++
""")|
Ln: 8 Col: 4
```

# Triple Quoted String

Prints exactly what is between the triple quotes.. even line feeds



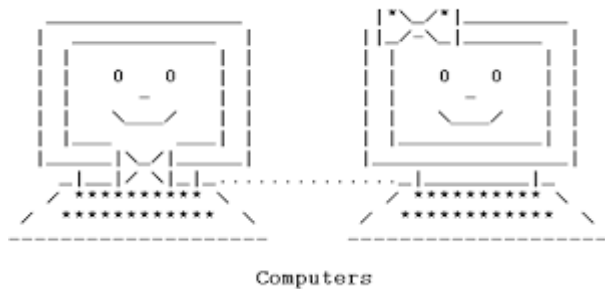
A screenshot of a Python 3.4.2 Shell window. The window title is "Python 3.4.2 Shell". The menu bar includes "File", "Edit", "Shell", "Debug", "Options", "Windows", and "Help". The main text area shows the following output:

```
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct 6 2014, 22:15:05) [MSC v.1600
 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> print("""
*****
*****
-----
+++++++
""")

*****
*****
-----
+++++++

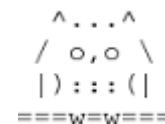
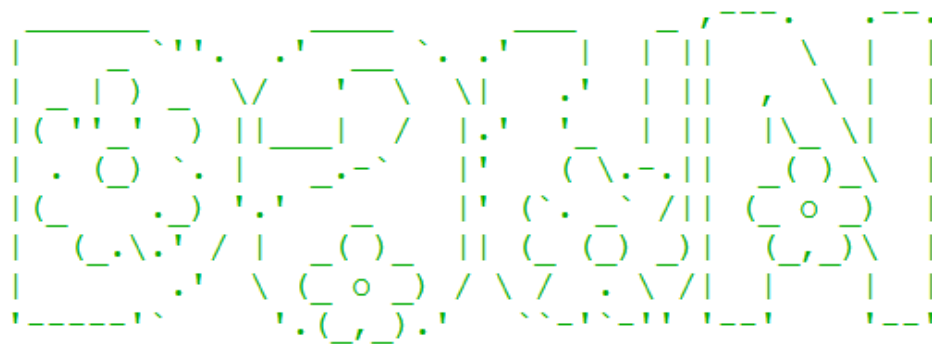
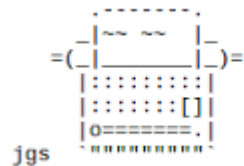
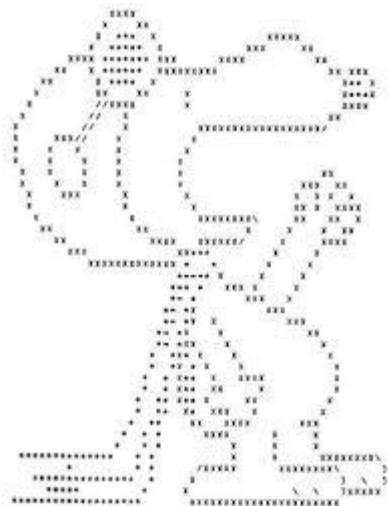
>>> |
```

Two pink arrows point from the left towards the first and second triple-quoted string literals in the code. The status bar at the bottom right shows "Ln: 15 Col: 4".



# Fun With ASCII Art

ASCII art is a graphic design technique that uses computers for presentation and consists of pictures pieced together from the 95 printable (from a total of 128) characters defined by the ASCII Standard

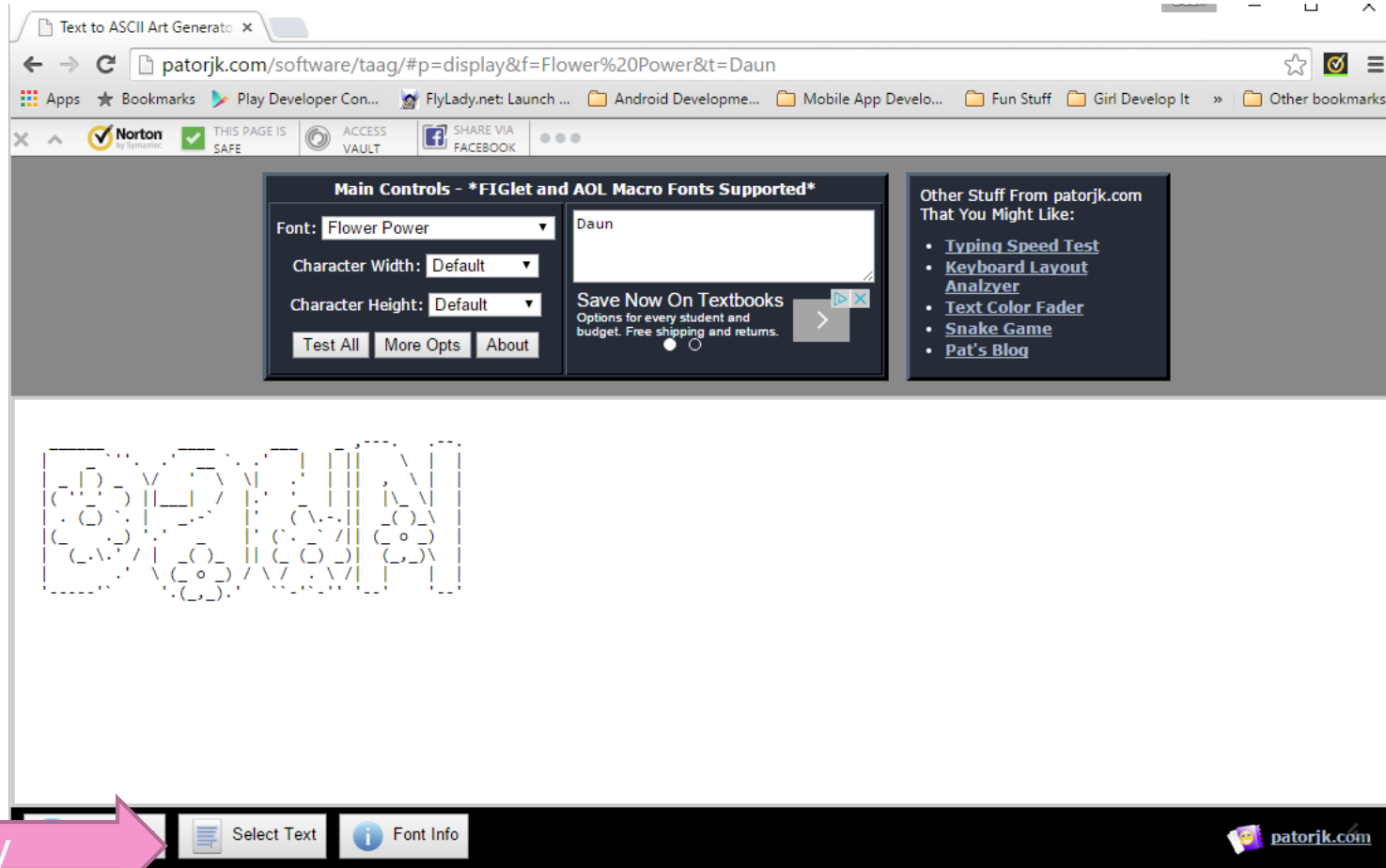


I am a great ASCII artist...  
Why are you laughing...?



# ASCII Art Website ([patorjk.com/software](http://patorjk.com/software))

Select this → **Text ASCII Art Generator** – A web app that lets you type in large ASCII Art text lettering. This can create art you can put in your email signature, on your webpage, etc etc.



To copy →

# Working Break Time



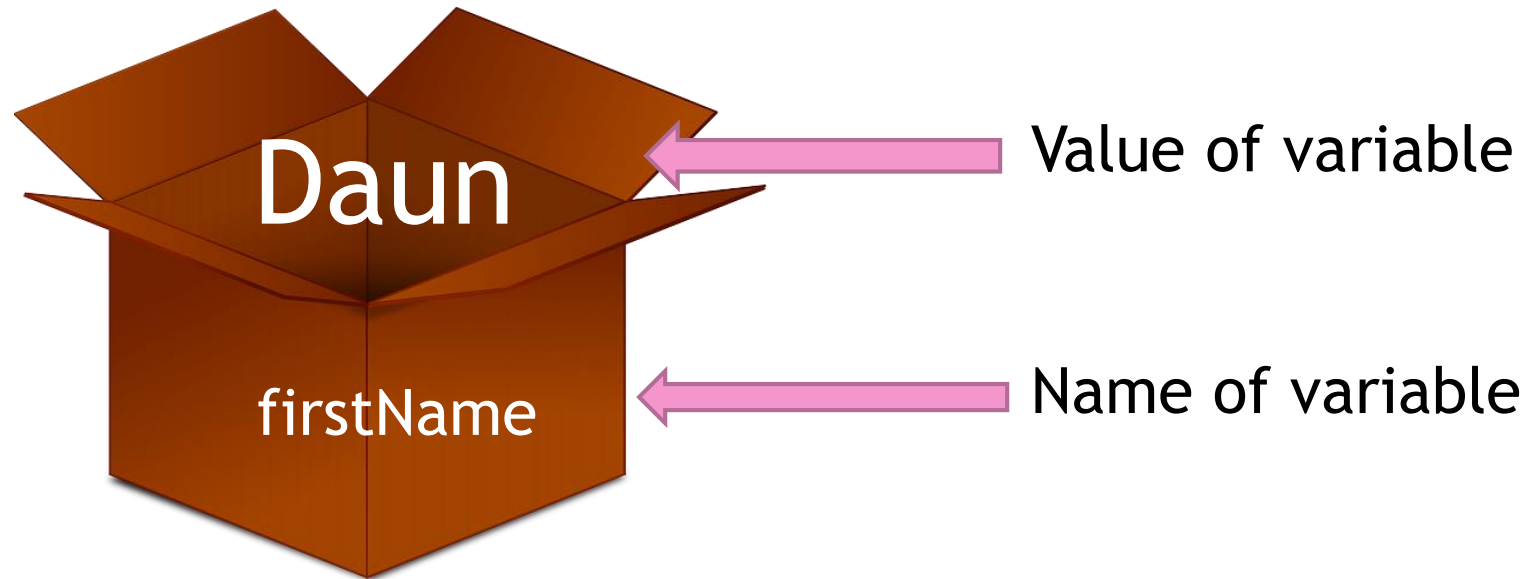
Use the ASCII Art Generator to create art from your name.  
Write a Python Program using triple-quoted strings to prints out your ASCII Art.  
(Optional) Have your program ding a bell when printing your name.

# Some Functions Return Values

We can “capture” these values and store them in the computer’s memory.

We store them by assigning the values to what is called a **VARIABLE**

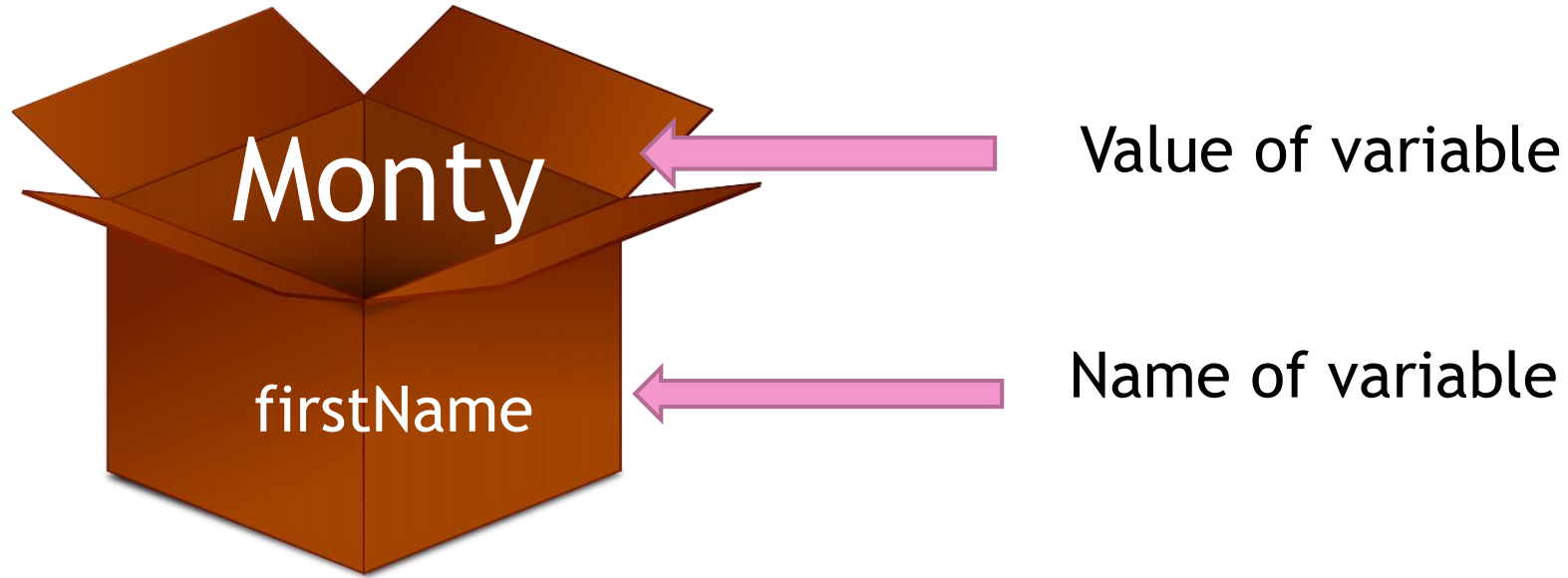
# Variables



**This variable holds a string**

`firstName = "Daun"`

# Variables

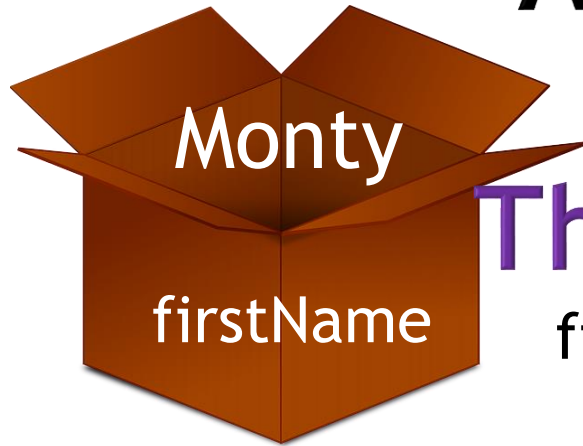


**This variable holds a string**

`firstName = "Monty"`

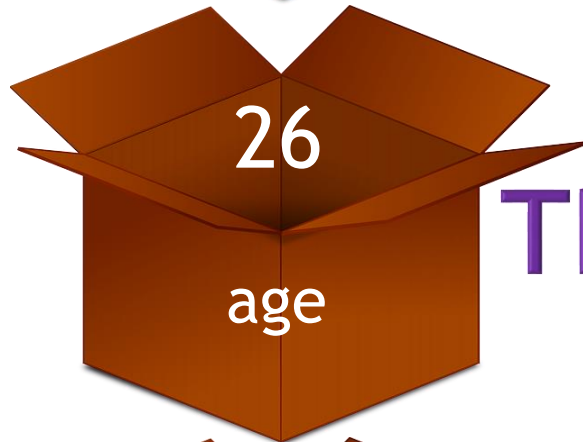


# Assign variables with =



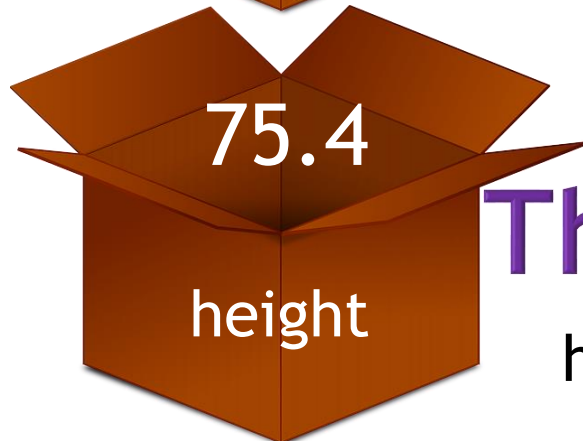
**This variable holds a string**

`firstName = "Monty"`



**This variable holds an int**

`age = 26`



**This variable holds a float**

`height = 75.4`

What type of variable does input() return?

```
firstName = input("Please enter your first name")
```

What type of variable does input() return?

```
firstName = input("Please enter your first name")
```

A String

How do we get an integer from `input()`?

How do we get an integer from input()?

**int()**

```
age = int(input("Please enter your age :"))
```

A function inside a function  
**TRICKY! ... Try it!**

How do we get an float from input()?

How do we get an float from input()?

**float()**

```
height = float(input("Please enter your height :"))
```

# How do we print 2 decimal places?

## `format(number, ".2f")`

```
File Edit Format Run Options Windows Help
height = float(input("Please enter your height :"))
print("Height is : ", format(height, ".2f"))
input("\n\nPress Enter to Exit.")
```

Rectangular Snip

C:\Python31\python.exe

Please enter your height :72.3

Height is : 72.30

Two decimals

Press Enter to Exit.



# Math

- = Assigns a value
- + Add
- Subtract
- \* Multiply
- / Divide

## Math

`twoYearsOlder = yourAge + 2`

`oneYearYounger = yourAge - 1;`

`evenNumber = number * 2`

`halfAge = yourAge / 2`

# Repeating Strings By Using \*

```
>>> print ("=+" * 20)
```

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
```

```
>>> print ("Hi!\n" * 6)
```

```
Hi!
```

```
Hi!
```

```
Hi!
```

```
Hi!
```

```
Hi!
```

```
Hi!
```

# Putting Strings Together (Concatenation)

File Edit Format Run Options Windows Help

```
string1 = "Daun"  
string2 = "Davids"  
print(string1 + string2)  
input("\n\nPress Enter to Exit.")
```

**+ sign**

C:\Python31\python.exe

Window Snip

DaunDavids

← No Space

Press Enter to Exit.

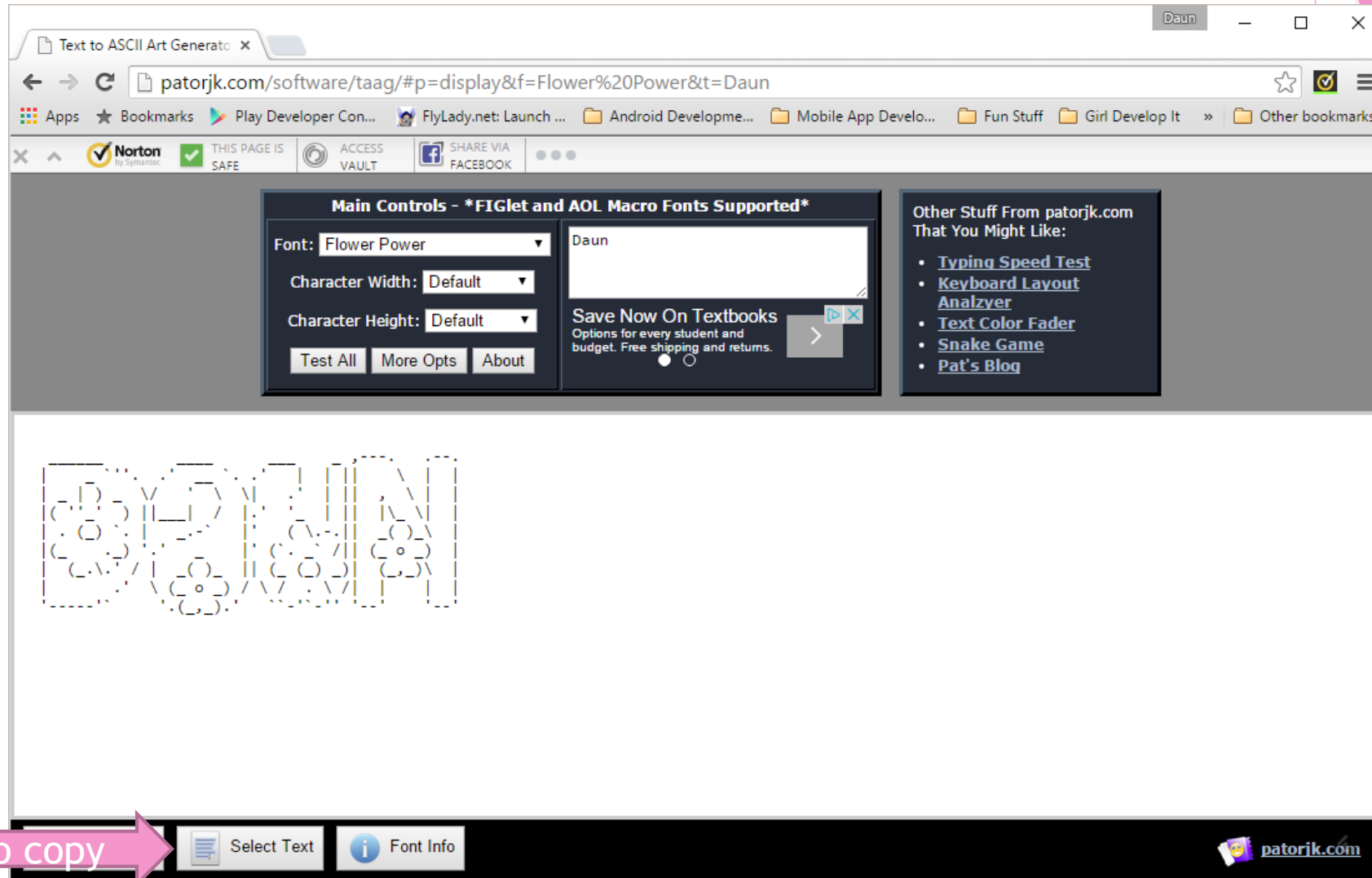
# Putting Strings Together (Concatenation)

```
ring.py - C:/Users/Daun/Desktop/Teaching/python/week 2/ring.py (3.4.2)
File Edit Format Run Options Windows Help
string1 = "Daun"
string2 = "Davids"
print(string1,string2)
input("\n\nPress Enter to Exit.")
|
```

## Comma

```
C:\Python31\python.exe
Daun Davids ← Has a Space
Press Enter to Exit.
```

# ASCII Art Website (patorjk.com/software)



# Triple Quoted String Stored as a Variable

File Edit Format Run Options Windows Help

```
title = |'''
```

The diagram illustrates a network of 10 nodes, each containing a mathematical expression. The nodes are arranged in a grid-like structure with various connections between them. The expressions involve square roots and fractions, often with subscripts or superscripts. The connections are represented by lines, some of which are labeled with '1' or '2'.

The nodes and their connections are as follows:

- Node 1 (Top Left):**  $\sqrt{\frac{1}{2} \left( \frac{1}{2} + \frac{1}{2} \right)}$
- Node 2 (Top Middle):**  $\sqrt{\frac{1}{2} \left( \frac{1}{2} + \frac{1}{2} \right)}$
- Node 3 (Top Right):**  $\sqrt{\frac{1}{2} \left( \frac{1}{2} + \frac{1}{2} \right)}$
- Node 4 (Middle Left):**  $\sqrt{\frac{1}{2} \left( \frac{1}{2} + \frac{1}{2} \right)}$
- Node 5 (Middle Middle):**  $\sqrt{\frac{1}{2} \left( \frac{1}{2} + \frac{1}{2} \right)}$
- Node 6 (Middle Right):**  $\sqrt{\frac{1}{2} \left( \frac{1}{2} + \frac{1}{2} \right)}$
- Node 7 (Bottom Left):**  $\sqrt{\frac{1}{2} \left( \frac{1}{2} + \frac{1}{2} \right)}$
- Node 8 (Bottom Middle):**  $\sqrt{\frac{1}{2} \left( \frac{1}{2} + \frac{1}{2} \right)}$
- Node 9 (Bottom Right):**  $\sqrt{\frac{1}{2} \left( \frac{1}{2} + \frac{1}{2} \right)}$
- Node 10 (Far Right):**  $\sqrt{\frac{1}{2} \left( \frac{1}{2} + \frac{1}{2} \right)}$

The connections between the nodes are as follows:

- Node 1 is connected to Node 2 by a horizontal line labeled '1'.
- Node 2 is connected to Node 3 by a horizontal line labeled '1'.
- Node 3 is connected to Node 6 by a diagonal line labeled '1'.
- Node 4 is connected to Node 5 by a horizontal line labeled '1'.
- Node 5 is connected to Node 6 by a horizontal line labeled '1'.
- Node 5 is connected to Node 8 by a diagonal line labeled '1'.
- Node 6 is connected to Node 9 by a horizontal line labeled '1'.
- Node 7 is connected to Node 8 by a horizontal line labeled '1'.
- Node 8 is connected to Node 9 by a horizontal line labeled '1'.
- Node 9 is connected to Node 10 by a horizontal line labeled '1'.
- Node 1 is connected to Node 4 by a vertical line labeled '2'.
- Node 2 is connected to Node 5 by a vertical line labeled '2'.
- Node 3 is connected to Node 6 by a vertical line labeled '2'.
- Node 4 is connected to Node 7 by a vertical line labeled '2'.
- Node 5 is connected to Node 8 by a vertical line labeled '2'.
- Node 6 is connected to Node 9 by a vertical line labeled '2'.
- Node 7 is connected to Node 10 by a vertical line labeled '2'.

|| || ||

```
print(title)
input("\n\nPress Enter to Exit.")
```

# Triple Quoted String





# What We Learned in Part 2

- \* Escape Sequences or How to print the unprintable!
- \* Triple Quoted Strings
- \* Fun With ASCII Art
- \* Variables
  - Strings
  - Integers
  - Floating Point Numbers
- \* Printing & Formatting Numbers
- \* Your Favorite Subject - MATH!

# Working Break Time



Write your own Python program that uses any ASCII Art stored in a variable.  
Google ASCII art generator to find programs that convert images into  
ASCII art.