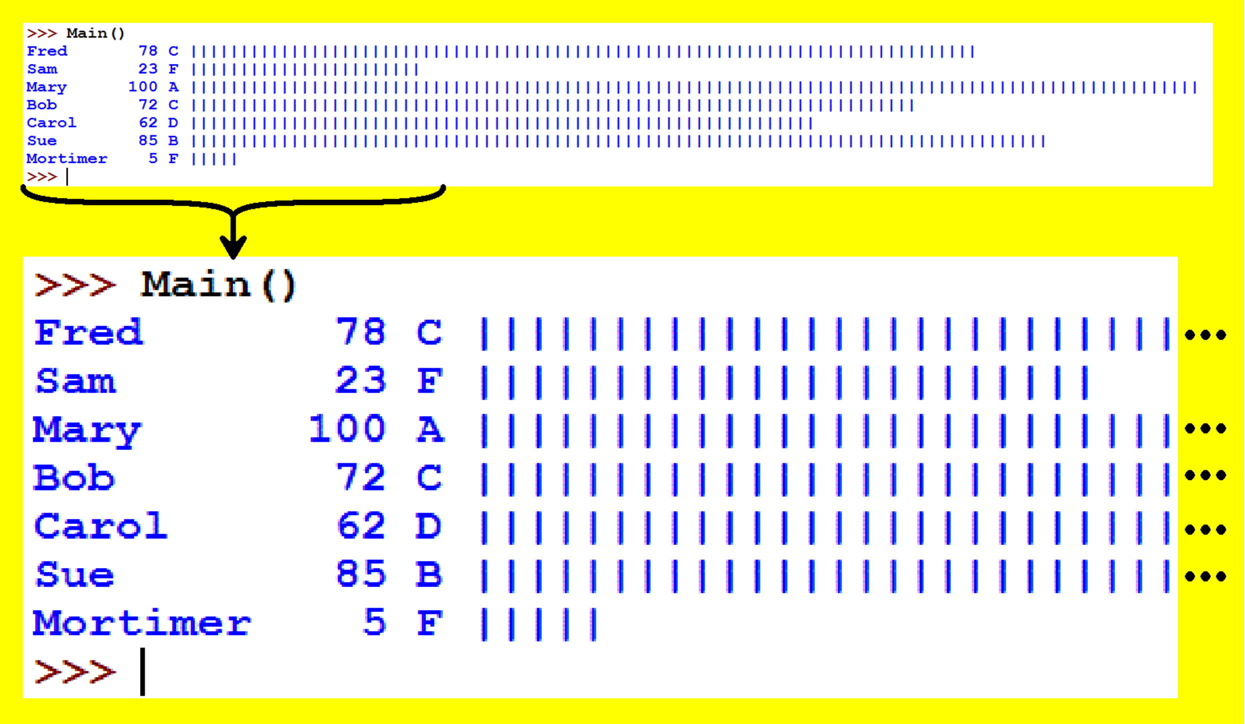
LAB #2 – Bar Graphs

The goal of this Python programming assignment is to successfully create, enter, debug, and run a simple program in the Idle environment, based on a given framework containing a main program and the stubs of four functions. You are to complete the code inside those four functions.

Background

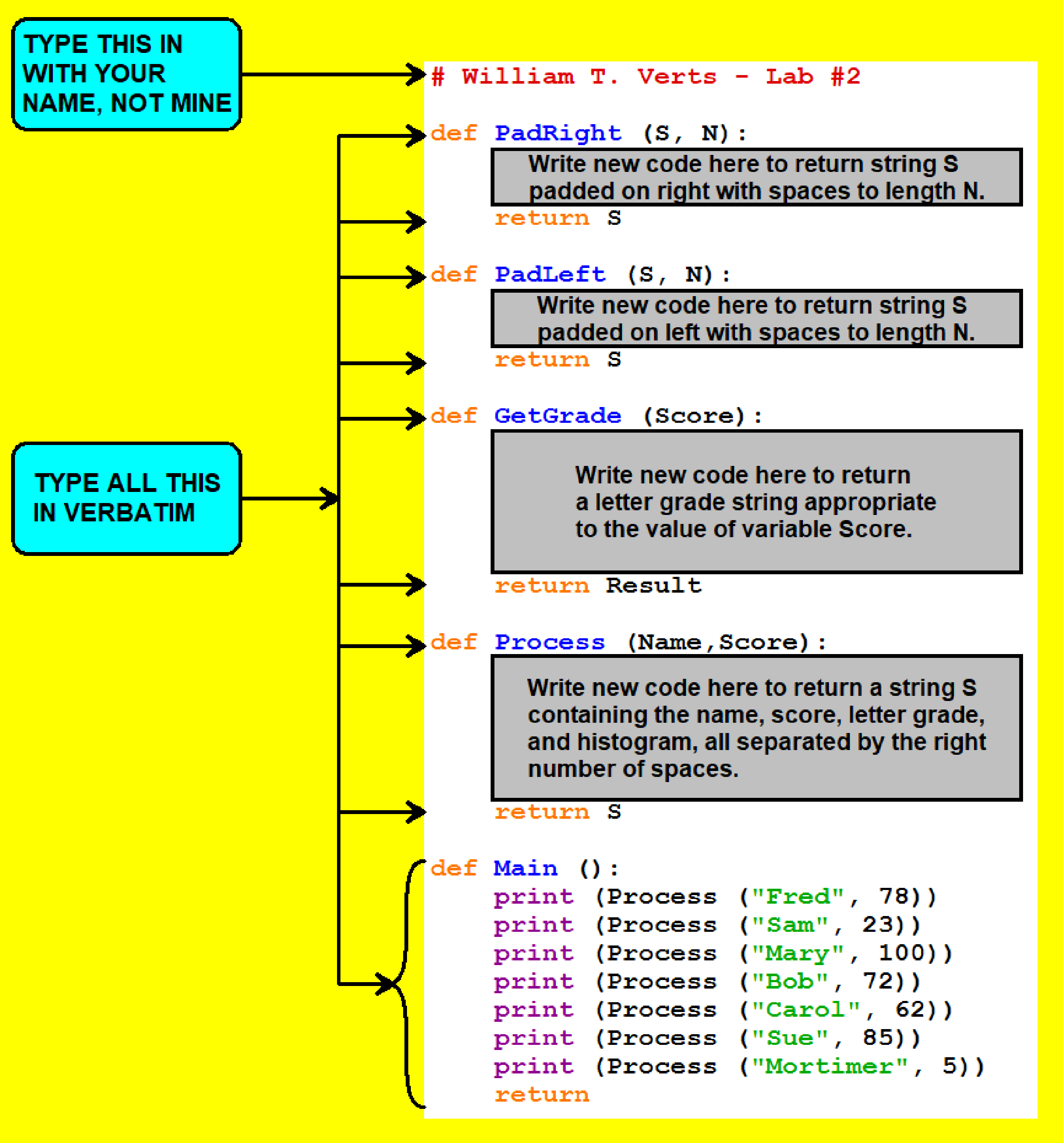
In this assignment you are going to write a program to enter a set of grades for students who have taken an exam, and then generate a set of text-based bar graphs based on those grades. For example, if student Fred received a 78 on his exam, the line of output for Fred will be his name, his score, his letter grade, and then 78 copies of the | character (the vertical bar), padded appropriately with blanks. When your program runs, the expected output from Idle/Python will look like the following image (the top image is the whole output, the bottom is the left part zoomed up so you can see it more easily):



```
>>> Main()
Fred       78 C ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sam        23 F |||||||||||||||||||||||
Mary      100 A ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Bob        72 C ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Carol      62 D ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sue        85 B |||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Mortimer    5 F |||||
>>> |
```

Setting up the Assignment

Type in the following program code framework ***exactly*** as you see it here, except with ***your name*** instead of mine, and then save it in your Python folder with `Lab2.py` as the filename. Leave the gray areas blank for now; you will write new code there later.



TYPE THIS IN WITH YOUR NAME, NOT MINE

TYPE ALL THIS IN VERBATIM

```
# William T. Verts - Lab #2

def PadRight (S, N):
    Write new code here to return string S
    padded on right with spaces to length N.
    return S

def PadLeft (S, N):
    Write new code here to return string S
    padded on left with spaces to length N.
    return S

def GetGrade (Score):
    Write new code here to return
    a letter grade string appropriate
    to the value of variable Score.
    return Result

def Process (Name,Score):
    Write new code here to return a string S
    containing the name, score, letter grade,
    and histogram, all separated by the right
    number of spaces.
    return S

def Main ():
    print (Process ("Fred", 78))
    print (Process ("Sam", 23))
    print (Process ("Mary", 100))
    print (Process ("Bob", 72))
    print (Process ("Carol", 62))
    print (Process ("Sue", 85))
    print (Process ("Mortimer", 5))
    return
```

Notice that there are four functions which need to be finished, `PadRight`, `PadLeft`, `GetGrade`, and `Process`. You can complete and debug each function in order, before working on the next function in the program. Function `Main` is finished as-is and ***must not be modified*** from what you see here.

Task #1 – PadLeft and PadRight

The `PadRight` function has two parameters: `S` (a string) and `N` (an int), and returns as its result the string `S` padded on the right with blanks until the length of `S` contains no fewer than `N` characters. The `PadLeft` function is identical to `PadRight` except that it adds blanks to the left side of the string.

For example, the string `"Frog"` is four characters long, so `PadLeft("Frog",7)` would return the seven-character string `"   Frog"` (with the extra blanks on the left) and `PadRight("Frog",7)` would return the seven-character string `"Frog   "` (with the extra blanks on the right).

However, the string `"Froggies"` is *already* longer than seven characters, so both `PadLeft("Froggies",7)` and `PadRight("Froggies",7)` would return the original string `"Froggies"` unchanged as the result.

`PadLeft("Frog", 7)`	**returns the 7-character string:**	□□□Frog
`PadRight("Frog", 7)`	**returns the 7-character string:**	Frog□□□

`PadLeft("Froggies", 7)`
`PadRight("Froggies", 7)`
	return the 8-character string:	Froggies

You will need to use a `while` loop in each function. You will also need to use the standard Python `len` function in your code to ask about the current length of a string, but no other functions.

Complete these two functions, then test them from the >>> command line prompt with different strings and different lengths to make sure that they handle all possible cases. Common problems include “off-by-one” errors, where the number of blanks might be one too few or one too many – check this!

Note that `PadLeft` and `PadRight` ***return a value*** but ***do not print*** anything! You will lose points if either of these functions use the `print` statement! All printing happens in the `Main` function.

Task #2 - GetGrade

The `GetGrade` function has one `int` parameter called `Score`. The function must return the one-character string `"A"` if `Score` is greater than or equal to 90, `"B"` if `Score` is greater than or equal to 80, `"C"` if `Score` is greater than or equal to 70, `"D"` if `Score` is greater than or equal to 60, and `"F"` otherwise.

Complete this function, then test it with different values for `Score` to make sure it handles all five grade ranges correctly, including exactly 90, 80, 70, and 60.

Note that `GetGrade` ***returns a value*** but ***does not print*** anything! You will lose points if this function uses the `print` statement! All printing happens in the `Main` function.

Task #3 - Process

The `Process` function has two parameters, `Name` and `Score`. Those parameters will be passed the name and the score for a particular student. For example, if the function is called as `Process("Fred",78)` then `Name` will contain the string `"Fred"` and `Score` will contain the integer `78`.

The `Process` function initializes a string `S` to the empty string, and then builds up and returns `S` as the value of the function, where `S` will contain the following items, in this order:

1: Student's name, padded on the right to 10 characters (columns 1…10),
2: Student's score, converted to a string, and padded on the left to 3 characters (columns 11…13),
3: Student's grade, padded on the left to 2 characters (columns 14…15),
4: one extra blank (column 16),
5: the correct number of vertical bar characters (columns 17…17+Score-1).

This function starts to build the string `S` by calling `PadLeft`, `PadRight`, and `GetGrade` with the appropriate actual parameters.

This function then also requires the use of a loop to add to the string the correct number of vertical bar characters; for this assignment I want you to use a counter loop (a variable with a `while`-loop, and not a `for`-loop).

Finish the function by writing code in the gray area to do all of this.

Note that `Process` ***returns a value*** but ***does not print*** anything! You will lose points if this function uses the `print` statement! All printing happens in the `Main` function.

Test your function by calling `Process` from the `>>>` command line prompt as `Process("Fred",78)` which would return as the value of `S` the following string:

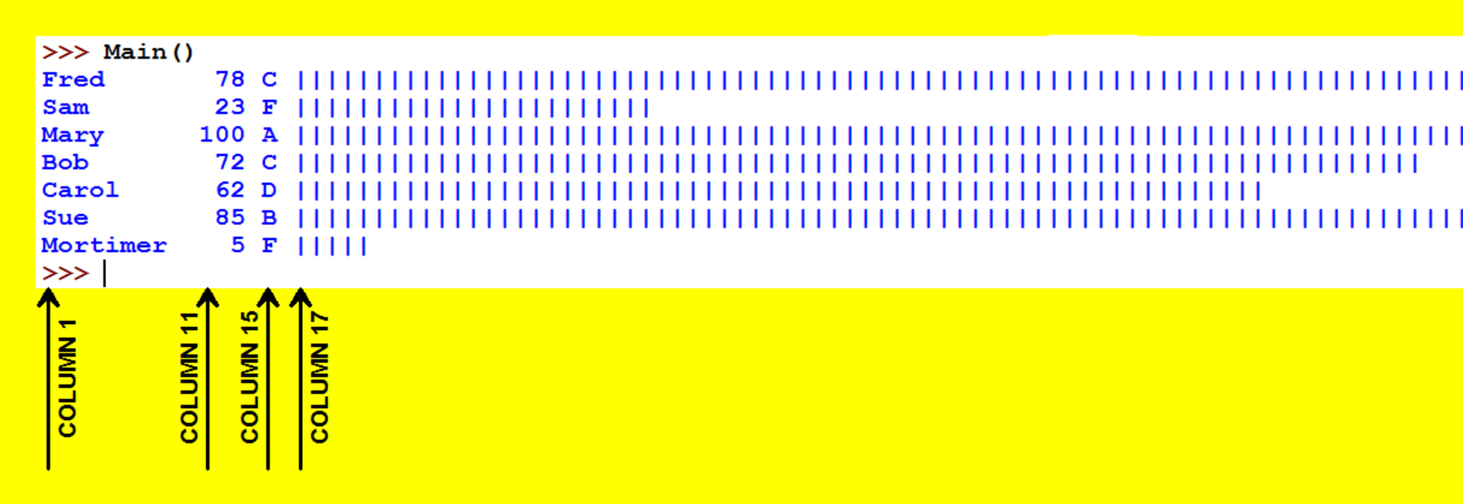
```
Fred       78 C ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
```

Similarly, manually test `Process("Sam",23)` and `Process("Mary",100)` to verify that your function works correctly. Common problems include "off-by-one" errors, where the number of vertical bars might be one too few or one too many – check this!

Running the Program

Run the program by typing `Main()` at the command prompt.

You should then get all seven lines of output as shown here, with the spacing as indicated in the following image:



```
>>> Main()
Fred       78 C |||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sam        23 F |||||||||||||||||||||||
Mary      100 A |||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Bob        72 C ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Carol      62 D ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sue        85 B |||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Mortimer    5 F |||||
>>> |
```

If the output does not contain the correct information, debug your program and run it again. Continue to edit and test your program until the resulting output does contain the correct information.

Finishing Up

When you are finished and everything runs correctly, go to the class site and click on the link for submitting lab assignments. In Idle select all the text, copy it to the clipboard, in the Web page paste the text into the program area of the submission form, fill in your name and ID number and the lab number in the appropriate slots, and then submit the assignment for grading as Lab #2.

The graders will score your program by running `Main` to see if it correctly generates the appropriate graphs. You will also be graded on efficiency as well as completeness, and the graders will be also testing each of your functions explicitly.

You will lose all credit for this assignment if you omit the comment containing your name at the top of the program.

You will lose all credit for this assignment if the program contains syntax errors that prevent it from running or if it crashes at any point before the execution is complete.

*You will lose **major** credit* if (a) you have changed the parameter list or return variable of any function, (b) if you have changed any of the framework code in the `Main` function, or (c) if you use the `print` statement in ***any*** of the four functions that you are to complete.

*You will lose **some** credit* if the spacing of the output lines is not ***exactly*** as specified in this assignment.