# BTE 601/ Fall18

***Assignment 4***

***Due on 10/2/2018 (11:59 PM)***

***(800 points)***

***This assignment has several parts (PLEASE READ CAREFULLY)***

***Main Assignment (800 Points)***

***PART I: You MUST do PART I (300 points). This is required even if you decide to do one of the alternate choices.***

***PART II: This is required unless you decide to do the one of the alternate parts.***

***PART III: This part is for extra credit – but you will only get extra credit if you have completed all parts of PART I and PART II and scored at least 90% on those two parts). No extra credit if you do the alternate options or you score less than 70% on the final exam.***

***Alternate Assignment (Maximum 300/500 for a max total of 600/800):***

***You may decide to do one of the alternate options instead of PART II at 60% of the score assigned to PART II in the main assignment. Your maximum score possible for the whole assignment will 600/800 if you choose this option. (300/800 from PART I and 300/800 from the alternate assignment option).***

***Please start early and be creative.***

## Please read carefully:

- ***Please be aware: I take academic integrity very seriously. If I suspect that you are giving me work that is not yours, you will get a zero until you come into my office and answer very in depth questions about the code. Don't risk it. I am very good at discovering this. You will do much better in the course this way. The final exam will be very similar to the problems you have here, and you will do it on paper. If you don't do this yourself, you will end up with a failing grade or a very low grade because your final exam will be literally blank because you have not practiced. I have given easy options if you can't do the more involved problems. No one gets an A or passes my courses by cheating. I guarantee it. It is not fair to those who are working very hard and putting the time in.***

- ***If you score less than 70% on the final exam, your maximum score on the assignment will be limited by the same percentage you scored on the final multiplied by the maximum possible score of the assignment. So, if you score 60% on the final exam, your maximum possible score would be 480/800. This does not apply to the alternative assignment part since it is already discounted. In such case the max possible score would also be 480/800.***

## **PART I (Required – Even if you decide to do the alternate assignment)** **(300 Points)**

This question is the same question from the previous assignment. Now, you must build the system using classes (OOP) and make it accommodate ordering from multiple restaurants each having its own menu. An order could only be from one restaurant.

You should at a minimum include the following classes:

*Restaurant, Menu, MenuItem, and Order.*

Each order should get a unique random ID that consists of alphanumeric characters, and the user should be able to save the order in a file with a filename that has the oder_xxxxxx_resturantname_MM/dd/yyyy_HHMMSS.txt.

Your initial display menu should give a list of restaurants available to order from.

You should have a file that looks something like this:

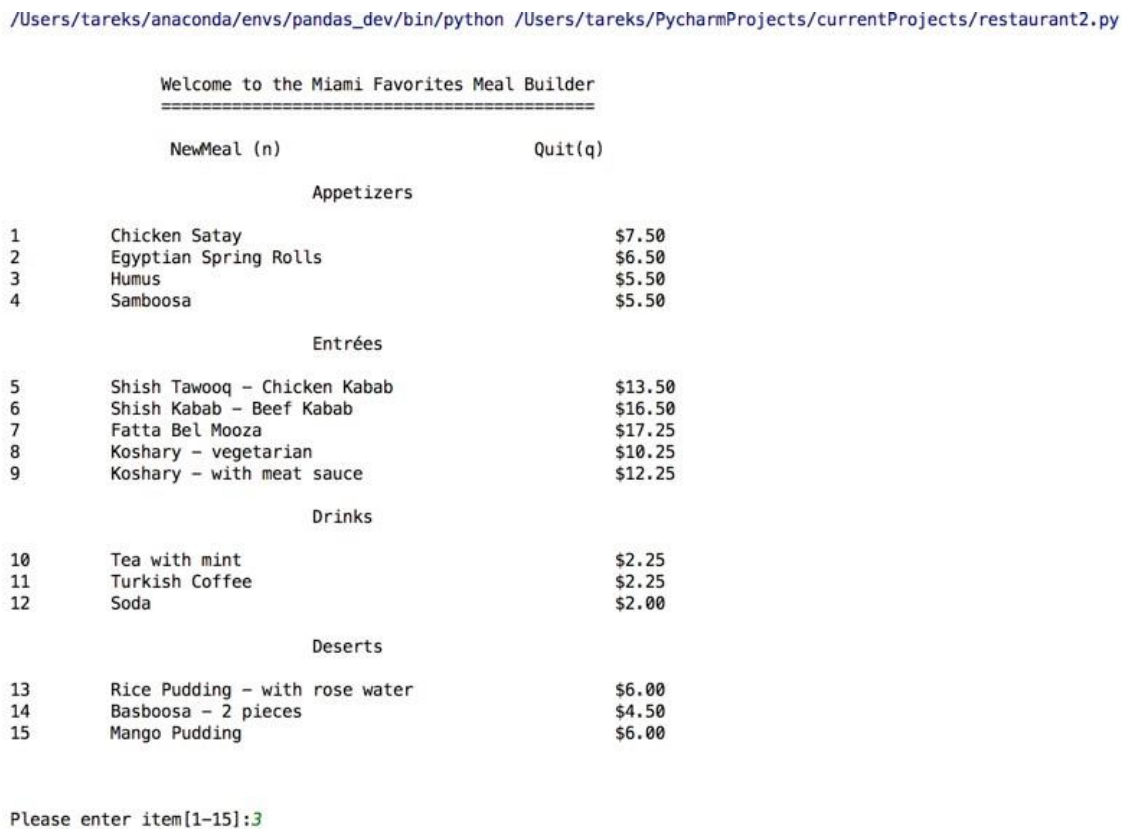
```
Restaurant1
Category, ItemName, price
Category, ItemName, price
Category, ItemName, price
END
Restaurant2
Category, ItemName, price
Category, ItemName, price
Category, ItemName, price
Category, ItemName, price
END
```

In this assignment, you will build a restaurant favorites Meal Builder. Your system will build the menu of any restaurant, and enable a patron to build a meal. The meal must include all categories (Appetizers, Entrées, Drinks, Deserts). You should make it dynamic to have more categories of course. Under each category, you will have menu items with pricing information. The system will enable a patron to build a meal (or several meals for multiple people), and price the whole ticket.

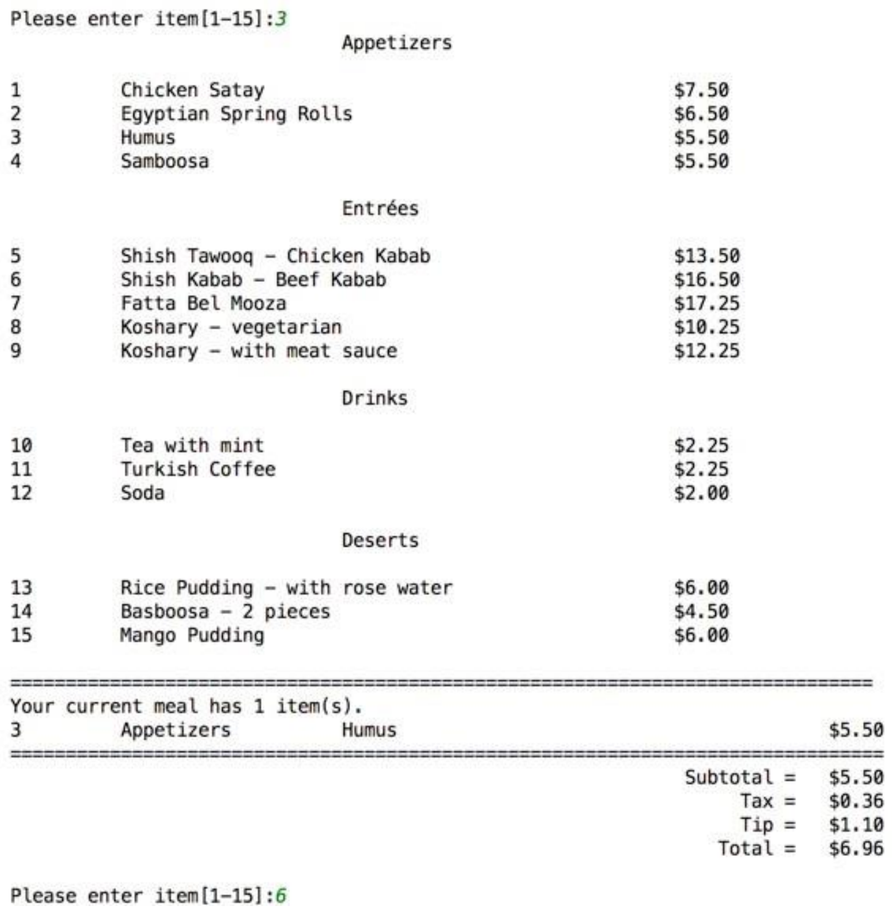
Please make sure you include options for the patron to quit or to start building a new meal. Your design can also be different from my implementation such that it would allow to delete an item from a meal – be creative.

Your output must be formatted and must produce a very elegant ticket. Here

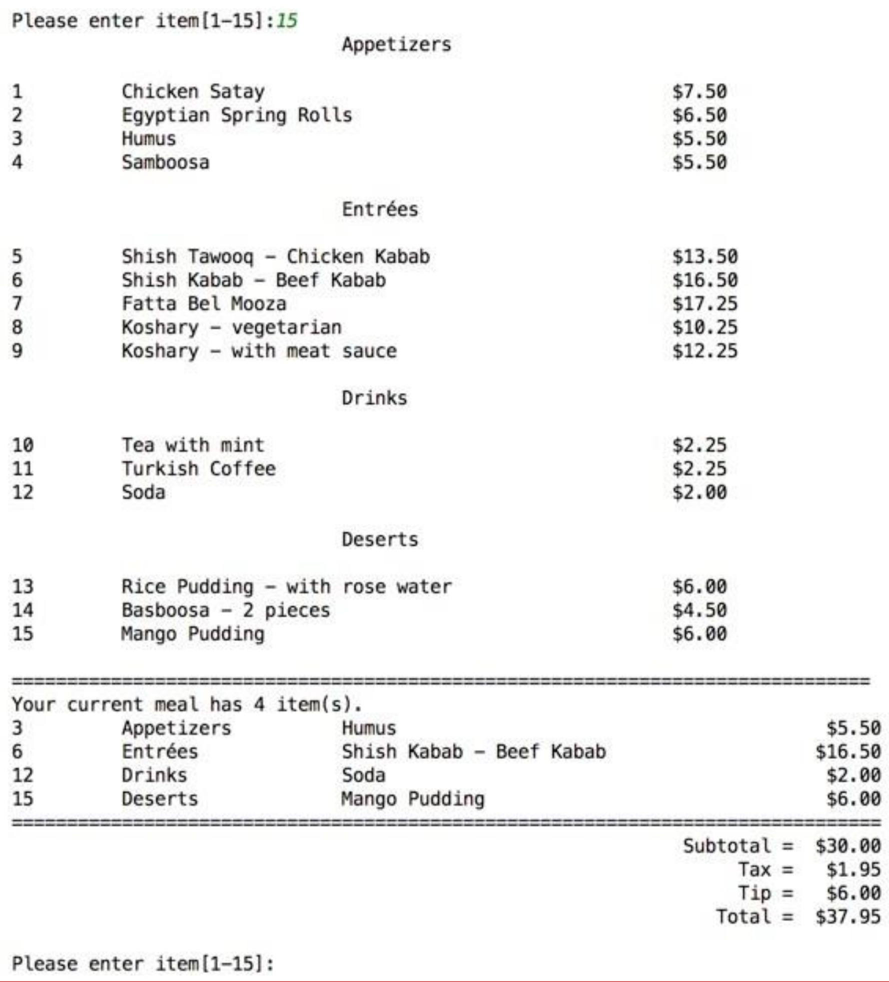
is an example implementation (the minimum functionality).



```
/Users/tareks/anaconda/envs/pandas_dev/bin/python /Users/tareks/PycharmProjects/currentProjects/restaurant2.py

              Welcome to the Miami Favorites Meal Builder
              ==========================================

               NewMeal (n)                          Quit(q)

                             Appetizers

1        Chicken Satay                                  $7.50
2        Egyptian Spring Rolls                          $6.50
3        Humus                                          $5.50
4        Samboosa                                       $5.50

                             Entrées

5        Shish Tawooq - Chicken Kabab                   $13.50
6        Shish Kabab - Beef Kabab                       $16.50
7        Fatta Bel Mooza                                $17.25
8        Koshary - vegetarian                           $10.25
9        Koshary - with meat sauce                      $12.25

                             Drinks

10       Tea with mint                                  $2.25
11       Turkish Coffee                                 $2.25
12       Soda                                           $2.00

                             Deserts

13       Rice Pudding - with rose water                 $6.00
14       Basboosa - 2 pieces                            $4.50
15       Mango Pudding                                  $6.00


Please enter item[1-15]:3
```



```
Please enter item[1-15]:3
                         Appetizers

1        Chicken Satay                                  $7.50
2        Egyptian Spring Rolls                          $6.50
3        Humus                                          $5.50
4        Samboosa                                       $5.50

                         Entrées

5        Shish Tawooq - Chicken Kabab                   $13.50
6        Shish Kabab - Beef Kabab                       $16.50
7        Fatta Bel Mooza                                $17.25
8        Koshary - vegetarian                           $10.25
9        Koshary - with meat sauce                      $12.25

                         Drinks

10       Tea with mint                                  $2.25
11       Turkish Coffee                                 $2.25
12       Soda                                           $2.00

                         Deserts

13       Rice Pudding - with rose water                 $6.00
14       Basboosa - 2 pieces                            $4.50
15       Mango Pudding                                  $6.00

==============================================================================
Your current meal has 1 item(s).
3        Appetizers         Humus                                       $5.50
==============================================================================
                                                           Subtotal =   $5.50
                                                                Tax =   $0.36
                                                                Tip =   $1.10
                                                              Total =   $6.96

Please enter item[1-15]:6
```



```
Please enter item[1-15]:15
                             Appetizers

1         Chicken Satay                                   $7.50
2         Egyptian Spring Rolls                           $6.50
3         Humus                                           $5.50
4         Samboosa                                        $5.50

                             Entrées

5         Shish Tawooq - Chicken Kabab                    $13.50
6         Shish Kabab - Beef Kabab                        $16.50
7         Fatta Bel Mooza                                 $17.25
8         Koshary - vegetarian                            $10.25
9         Koshary - with meat sauce                       $12.25

                             Drinks

10        Tea with mint                                   $2.25
11        Turkish Coffee                                  $2.25
12        Soda                                            $2.00

                             Deserts

13        Rice Pudding - with rose water                  $6.00
14        Basboosa - 2 pieces                             $4.50
15        Mango Pudding                                   $6.00

==============================================================================
Your current meal has 4 item(s).
3         Appetizers          Humus                                   $5.50
6         Entrées             Shish Kabab - Beef Kabab               $16.50
12        Drinks              Soda                                    $2.00
15        Deserts             Mango Pudding                           $6.00
===============================================================================
                                                       Subtotal =  $30.00
                                                            Tax =   $1.95
                                                            Tip =   $6.00
                                                          Total =  $37.95

Please enter item[1-15]:
```

## PART II
## (500 Points)

In this assignment, you will be performing some important data-processing operations, specifically sorting a large database file. Sorting data is a very important operation in computing for many reasons. One of those reasons is that it makes the data more accessible to humans once it is printed (imagine trying to use a telephone directory in which the names do not appear in any particular order). Another reason is that it makes the data more quickly searchable by the computer.

There are many large data files to use for this assignment, but you will only need the first one until you get on to the advanced parts. They are all available on blackboard, and are named people1.txt, people2.txt, people3.txt, people5.txt, people10.txt, people20.txt, people30.txt, people50.txt, and people100.txt. ***You must represent a person as an object of class PersonType, throughout the assignment.***

Look at the file "`people1.txt`" with a text editor. You will see that it contains data about a number of people. Each line contains exactly five items: a person's social security number, their first name, their last name, their date of birth, and state of residence. The five items are separated by spaces, but no item will ever contain a space. Here is a sample from the middle of the file:

```
320990814 Arthur Farmer 19560424 NV
322230050 Eros Crandon 19250819 TX
324640114 Lusitania Lissom 19440104 IN
325400784 Rose Terwilliger 19260122 WI
327640597 Jeffrey Stone 19760801 DE
327950765 Mary Emmett 19290224 CO
328610085 Heironymous Inchworm 19661102 CA
329310410 William McCormick 19550819 WV
329320248 Nicola Birchmore 19230107 IA
330270343 Pauline McTaggart 19290402 MN
331130693 Jim Trombone 19411222 NE
331960453 Abraham Larch 19750901 WY
332040687 Trixie Underwood 19200516 UT
```

As you may have noticed, the date of birth is provided as a single integer, in the format `yyyymmdd`; Arthur Farmer was born on the 24th of April 1956. The `1` in the filename `people1.txt` indicates that it contains exactly one thousand lines.

1. *Read the Data*

Write a program that creates a list of type **PersonType** to read all the data from the file into that list. Of course, it will have to be a list of type Person that you will also need to define. Make your program close the file, then print out the first 10 items of data from the list, so that you can make sure everything was read correctly. You can use intermediate lists of lines while you are constructing the persons list if you would like.

2. *Basic Search*

Make your program ask the user to enter a name. It should then search through the data in the list (don't read the file again), finding any entry with a matching name. Correct matches with either first or last name should be accepted. For every matching entry that is found, print out all four data items: the social security number, first and last names, and date of birth of each matching person.

Remember that if you use the == operator to compare strings, the test is case-sensitive. The user (i.e. you) will have to type the name exactly correctly, with capital letters in the right places.

Important: Good clean design will make this lab much easier. Write a separate function that searches the list, do not put all the work in main.

3. *Find the Oldest*

Modify your program so that after closing the file, instead of printing the first ten items of data, it searches through *all* of them to find the oldest person represented. It should print the social security number, first and last names, date of birth, and state of the oldest person found.

Important: As for part two, good clean design will make this lab much easier. Write a separate function that searches the list to find the oldest person, do not put all the work in main.

4. *Promote the Oldest*

For some unfathomable reason, the management wants the oldest person to occupy the first position in the list. Modify your program so that after finding the oldest person, it swaps his or her data with the data already occupying the first position in the list. Remember that the first position in a list is numbered zero, not one.

5. *Now Promote the Second Oldest.*

The management has now decided not only that the oldest person must occupy the first position in the list, but also that the second-oldest person must occupy the second position in the list. So, after searching for the oldest and moving their data to the front of the list, now search the remainder of the list (all except the first element), and move the oldest person you find (which must be the second oldest of all) into the second position of the list. Make sure you swap data, so that whoever was originally in the second position is not lost.

6. *More of the Same.*

The management are going to keep on adding requirements like this, next putting the third-oldest in the third position, then the fourth, then the fifth. There is no knowing when they will grow out of this petty obsession, so make things easier for yourself. Modify your search function so that it can be told how much of the list to search. That is, give it two `int` parameters (let's call them `a` and `b`); its job is now to search only the portion of the list between position `a` and position `b`, to find the oldest person therein. This makes it very easy to search the remainder of the list to find the second and third oldest.

7. *The Ultimate Demand.*

Now the management make their final demand. You are to repeat the process of moving the nth-oldest person into the nth position 1000 times. (please remember, 1000 is the number of data records in the whole file).

This will result in the list being completely sorted. Do it, and check that it worked. Make your program print the contents of the list after it has finished. Look at the output to make sure that everyone is printed in order of their age.

Try to implement your own ***selection sort*** function – instead of using the Python sort.

(Research item/Optional – extra credit) Try again with bubble sort, insertion sort, quick sort, and compare the results in terms of the execution time of the sorting of records.

8. *Sorting the File.*

Once you have sorted the contents of the list, it might be a good idea to save the sorted data in a file. Make your program create a new file, and write all the contents of the list into that file in a sensible format. Use a text editor to look at the file and verify that it has the same format as the original file, and all the data is properly sorted.

9. *How Fast Is It?*

It is important to know how long computer operations are going to take when they have to work on a large amount of data.

Use a function (twice) to time how long it takes the computer to sort the list of 1000 data items. Do not include the time it takes to read the file or the time it takes to write the new file, just the pure sorting time. Note the time that you observe.

Now you know how long it takes to sort a database of 1000 items. How long do you think it would take to sort a database of 2000 names? 3000 names? 10,000 names?

Think about those questions, and work out what you believe the answer is. Then find out what the real answer is. The other files have exactly the same format as people1.txt, but are longer. `People`*N*`.txt` contains N thousand data records. If your program was nicely written, it will be a few seconds' work to change the list size and make it read a different file.

See how long it takes to sort these larger files, and compare the results to your predictions. If your predictions weren't substantially correct, make sure you understand why. You have just demonstrated a very important phenomenon of computing.

10. *Friendships (150 points)*

    a. Copy your code to a separate program.
    b. You will work with the list of persons.
    c. Implement or use a function random_in_range(int a, int b) function to choose a random integer number between two integers.
    d. Add a friends list for your PersonType class.
    e. For each person in the person list choose a number of friends (minimum is zero and maximum is Size/200). Assume this number is x. Now you need to generate x locations of the friends and add them as friends to the list of friends of the current person.
    f. Write or use a function that will sort the list now by the number of friends in a descending order.
    g. For each person, find other people who have common friends with that person and who those common friends are.
    h. For each person, find all the people who have that person as a friend.
    i. Repeat part e to randomly unfriend people (unfriending is zero to half of the number of friends).
    j. Repeat part f after you have run the unfriend part.

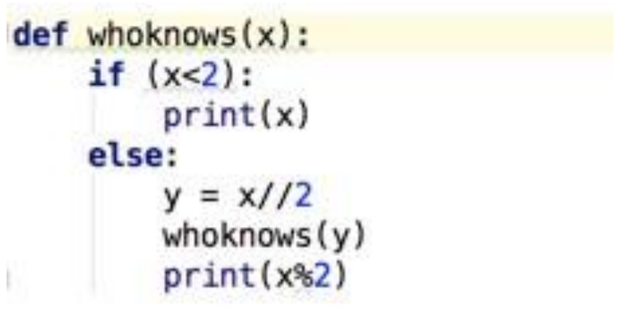
11. *Analytics (50 Points)*
    a. *For each year in the set, find the number of people who were born in that year.*
    b. *Find the number of people who reside in each state. Sort the states by the number of people from the set residing in each state in a descending order.*
    c. *Find the most common first name.*
    d. *Find the frequency of each first name. Sort the frequencies in a descending order.*
    e. *Find the most common last name. Sort the frequencies in a descending order.*
    f. *Make bins of age groups with each bin representing a 10 year period (you need to do this programmatically), and find the number of people in each bin.*
    g. *Now graph a histogram using the number of people for each age bin.*

12. *Menu System* ***(Extra Credit 50 points)***
    a. *Make Your System menu driven. Your menu should include options to add a person, search for a person by last name, display the friends of a person, add a friend, delete a friend, and perform all the analytics functions.*

## PART III (Extra Credit 60 Points – Does not apply to the alternative assignment)

***No extra credit unless you get a full mark on the previous two parts.***

1. (a) Explain in English what the following function will do. Explain how it works.



```
def whoknows(x):
    if (x<2):
        print(x)
    else:
        y = x//2
        whoknows(y)
        print(x%2)
```

(b) What will be the output if the following calls are
made: whoknows(2) =
whoknows(15) =
whoknows(-3) =

(c) Write a function *digitize* (using loops) that takes two parameters: one integer parameter and one bool parameter. The function would print the integer one digit at a time each on a separate line. If the bool parameter passed were true, the function would print the digits from the most significant digit to the least significant. Otherwise, it would print it in the reverse order (least significant to most significant). Don't use python built in functions. Use math logic with division and %.

| *Function Call* | *Output* |
|---|---|
| `digitize(1758,true)` | 1 7 5 8 |
| `digitize(1758,false)` | 8 5 7 1 |

(d) Write a function (without using loops) that reverses the digits in an integer and prints out the integer in this reverse form. It is not necessary to calculate the value of the reverse integer, just print out the digits in reverse order. The function should be called *reverse*. Remember to explain your functions, either by adding comments or using pseudocode or showing how you derived the function. State any assumptions you make.

2. (a) Write a function, *printdivisors*, that takes a single integer parameter and

prints all the numbers less that the parameter that are divisors of the parameter (i.e. divides it without a remainder) including 1. So *printdivisors(6)* will print 1,2,3. Note you may use a wrapper function or default parameters.

(b) Write a function, *sumdivisors*, that takes a single integer parameter and returns the sum of all the divisors of the parameter (including 1). So *sumdivisors(6)* will return 6 as 1+2+3=6. Note you may use a wrapper function or default parameters.

(c) Write a function, *allperfects*, that takes two parameters, each an integer, in any order and prints out all the perfect numbers between the lower parameter and the higher parameter. A perfect number is one is which the sum of its divisors is equal to the number itself.

Remember to explain your functions, either by adding comments or showing how you derived the function. State any assumptions you make

3. (a) Write a recursive function, *printZeros,* which prints out a series of zeros. The function takes one parameter and prints out the number of zeros specified by the parameter. So *printZeros(4)* will print: 0000 and *printZeros(2)* will print 00.

   (b) Write a recursive function, *printZPattern*, which prints out a pattern of zeros as follows:

   *printZPattern(3)* outputs:
   ```
   000

   00
   0
   ```
   *printZPattern(1)* outputs:
   ```
   0
   ```
   *printZPattern(4)* outputs:
   ```
   0000

   000
   00
   0
   ```

   (c) ow would you modify your second function to print a mirror pattern, such as (you do not have to code this one, just explain):
   *printZPattern2(3)* outputs:
   ```
   000
   00
   0
   00
   000
   ```

   Remember to explain your functions, either by adding comments or showing how you derived the function. State any assumptions you make.

4. Suppose you have a function, *myfib*, declared as follows:

   *def myfib( n):*

   which returns the nth Fibonacci number, using this function:

(a) Write a function, *myfibseries*, that also takes a single parameter, *n*, and prints out the entire sequence of Fibonacci numbers up to and including *n*, with a comma between each (do NOT use loops).

(b) Write a function, *myfibseriesR*, that also takes a single parameter, *n*, and prints out the entire sequence of Fibonacci numbers up to and including *n*, with a comma between each, but in REVERSE order (do NOT use loops).

(c) Write a function, *myfibsum*, that takes a single parameter, *n*, and prints out the sum of all the Fibonacci numbers up to and including *n* (do NOT use loops).

5.

(a) Write a program that reads 10 numbers from the user as input and loads them into a list.

(b) Add a function ***search*** that takes one int parameter –n- and searches for all the occurrences of n in the list.

6. Guessing Game(max of 3 tries): Write a program to simulate the following guessing game using the *random_in_range* function :(Basically, the program should generate a secret number in the range the use specifies and it guides the user until she gets it right or exhausts 3 attempts.

*The output of your program should be something like this:*

```
Enter min of range
10
Enter max of range
20
Guess a number between  10 and 20
Guess a Number:
17
You went too low
Guess a Number:
18
You got it!
```

## Alternative Assignment (Maximum 60% of the total score)

**Please do ONLY ONE of the following two options**

**Option I**

In this question you are required to use Python to solve a problem of your choice. This could be a finance, accounting, science or any business, engineering or medical application. A game of your choice is ok too (no Graphical User interface would be required). If you are interested in writing a multi-user game, come see me and I will guide you or look up how sockets could be defined – there are many examples on the web. Of course, you can look up how to do things, but you need to use YOUR OWN CODE. **Your program should include:**

1- Classes (use hiding, getters and setters, multiple constructors and methods).
2- Use functions – make your main very short. Try not to use any long detailed logic in main and modularize your application.
3- Input and Output files.
4- Make your own data set. You can even use random functions to create data sets or use a data set that would be deemed appropriate for your application.
5- Use static variables in classes so that you would give an entity an auto-incremented ID as needed.
6- When using a data set, make sure you demonstrate your knowledge of sorting and searching (no need to prompt the user if you don't want to, but feel free to design your application as you deem appropriate).
7- For a very large extra credit, get a raspberry pi, configure it, and run the application on it.
8- You must use some containers (lists and/or maps).
9- You can use recursion if you know how and if you need it.
10- You may use any of the functions I have on BB in the code or from the book.

Come up with something cool or something useful. It does not have to be long but it must show what you learned in the course. Please make sure it is all yours from the idea to the implementation. I made this so open ended so that you gauge it to how much you want to do and you are able to do – but it MUST be yours – don't copy – it will not work. Come up with your own idea. Of course, no two ideas will come out similar, and if they do, the implementations will be completely different. Also, this must be something that has significance – as an idea and a prototype implementation. It can't be something trivial. In other words, don't give me a hello world program and expect that you will get a grade for it

**Option II**

Write a 5-6 page paper (in Latex https://www.sharelatex.com (Preferred)) about **ONE** of the following topics (It must be in your own words):

1- Compare Python with Java, C++ and Ruby and how programming languages and tools are evolving (cloud, mobility, IoT, features, enhancements, complexity, performance, openness, etc), and when the use of one would be more appropriate than the others (if you such preference exists).
2- C++ 11 and C++ 14 features and enhancements – you must explain in your own words – don't copy. Do the research. How does this make C++ compared to Python, Java and Ruby? Do you think using boost libraries place C++ ahead of those languages when it comes to abstraction? Why is C++ is still so popular in some settings? What are those settings and when would you use it over any of the other languages above?
3- Advances in AI – Technical Perspective. What is new? What are the recent technical advances and challenges? How the technical landscape of Computational Sciences is well suited for yet another leap of advancements in Artificial Intelligence research and capabilities?
4- Functional Programming – with examples of how different modern languages support functional programming – with examples of your own.
5- The Internet of Things Development - You can discuss the different tools and platforms or you can discuss IoT applications in a specific domain (business, engineering or medical).
6- Cloud computing Development platforms (any of these topics: architecture, possibilities, scale, application – why it makes business sense, etc). You will need to discuss what services such as Azure, AWS or Bluemix offer and how this is revolutionizing software development. You can also approach this from an IaaS, PaaS, SaaS perspective but you need to discuss the stack of tools and the architecture along with some applications – possibly in a related domain (use of a business case scenario is perfectly fine).
7- Social Media Analytics – benefits, and what it means to business.
8- Big Data, Data Mining and Machine Learning – you could be general or use industry specific examples – but you need to discuss tools and architecture.
9- **Any other topic** – It must be related to programming, application development or specific technologies in a specific domain.

You must appropriately cite your sources in appropriate format. You can use IEEE format or LNCS format for your paper – or you can choose another format as you deem appropriate.

***Submission Instructions***

- ***Submission – code pasted in a word document with screenshots for every output. No screenshots of output will get a zero score for the question. You must submit before the deadline on BB.***
- ***Please submit on time and the right way- check your documents before you submit.***
- ***You must submit what you have. I am not going to chase you to submit your work. I will submit the final grade without.***
- ***Please be aware: I take academic integrity very seriously. If I suspect that you are giving me work that is not yours, you will get a zero until you come into my office and answer very in depth questions about the code. Don't risk it. I am very good at discovering this. You will do much better in the course this way. The final exam will be very similar to the problems you have here, and you will do it on paper. If you don't do this yourself, you will end up with a failing grade or a very low grade because your final exam will be literally blank because you have not practiced. I have given easy options if you can't do the more involved problems. No one gets an A or passes my courses by cheating. I guarantee it. It is not fair to those who are working very hard and putting the time in.***

- ***If you score less than 70% on the final exam, your maximum score on the assignment will be limited by the same percentage you scored on the final multiplied by the maximum possible score of the assignment. So, if you score 60% on the final exam, your maximum possible score would be 480/800. This does not apply to the alternative assignment part since it is already discounted. In such case the max possible score would also be 480/800.***