# Introduction to Computer Science and Programming 1

# CSCI120

### Chapter12: Classes

Assignment

|  |  |  |
| --- | --- | --- |
| **# of Students in the Group:** |  | |
|  |  |  |
| **Student 1** | *First name, last name* | *Student-ID* |
| **Student 2** | *First name, last name* | *Student-ID* |
| **Student 3** | *First name, last name* | *Student-ID* |
| **Student 4** | *First name, last name* | *Student-ID* |

# Requirements

* Please use meaningful name for your variables and functions
* Try to reuse your solutions as much as possible.

# Problem0:

# Define a class and a static method that receives a string (word) and reverses it and returns it.

**Problem1:**

* Design a class that represents a histogram like the one in the image below. The X axis represents the days of the week and then the Y axis represents the sales.
* Add an instance method which receives a day and return the corresponding Y value.

Chart, bar chart

Description automatically generated

**Problem2:**

* Define and design a class with a static method called *convert*, which receives a positive number and a base number and converts the number to the given base and returns it as a string.
  + For instance: convert (9, 2) returns 1001

# Problem3:

* Look around your classroom or your bedroom and come up with 3 classes that you see there.
* For each class define 3 instance Variables
* For each class define 1 static variable
* For each class define an accessor and a mutator for each instance variable
* For each class define 2 instance methods in addition to the accessors and the mutators.
* For each class define a constructor which initializes all its instance variables.

# Problem4:

* Write a main function and inside the function write a test scenario for each of the above classes. Your test scenario should include creating one or two instances of each class, call some of its accessors and mutators and its instance variables.
* Example:

**class Student:**

\_studentIDCounter = 1000

def \_\_init\_\_(self, firstName, lastName, address, age):

self.\_listOfCourse = []

self.\_firstName = firstName

self.\_lastName = lastName

self.\_address = address

self.\_age = age

Student.\_studentIDCounter = Student.\_studentIDCounter + 1

self.\_studentID = Student.\_studentIDCounter

def \_getFirstName(self):

return self.\_firstName

def \_getLastName(self):

return self.\_lastName

def \_getAddress(self):

return self.\_address

def \_getAge(self):

return self.\_age

def addCourseGrades(self, grade):

self.\_listOfCourse.append(grade)

def calculateAverage(self):

if len(self.\_listOfCourse) > 0:

sum = 0

for item in self.\_listOfCourse:

sum = sum + item

average = sum / len(self.\_listOfCourse)

return average

else:

return

def getStudentID(self):

return self.\_studentID

def printStudentProfile(self):

print("============================")

print(self.\_firstName)

print(self.\_lastName)

print(self.\_age)

print(self.\_address)

print(self.getStudentID())

print(self.calculateAverage())

**class TesStudent:**

def test(self):

student1 = Student("Peter", "Mak", "Vancouver", "29")

student2 = Student("David", "Cameron", "Burnaby", "30")

student1.printStudentProfile()

student2.printStudentProfile()

student1.addCourseGrades(80)

student1.addCourseGrades(73)

student1.addCourseGrades(85)

student2.addCourseGrades(48)

student2.addCourseGrades(90)

student1.printStudentProfile()

student2.printStudentProfile()

def main():

testStudent = TestStudent()

testStudent.test()

main()

# Problem5:

* Write a python program with the following description:
* Define a class called MyCustomList.
* The class has an instance variable called myList which is a list of integers.
* Define the following instance methods for the class:
  + addItem: It has an input of type int and it add the number to the list if this number already does not exist in the list. If it exits it will ignore it.
  + calculateSum: It has no input and will return the sum of all numbers in the list.
  + calculateMax: it has no input and will return the maximum number of the list.
  + printList: It has no input but print the current content of the list to the out.
  + Remember to define a constructor for this class.
* Define another class called Test MyCustomList. This class has a static method called testMyCustomList. This method is used to contain a test scenario for the MyCustomeList.
* In another python file, define a main function and use the TestClass to test the MyCustomeList class.

**Good Luck ☺**