Paper coding

Try to fully understand the basic concept before moving on to the next step.

Lack of understanding basic concepts will increase your burden in learning this course, which may make you fail the course.

It may be difficult now, but for successful completion of this course we suggest you to fully understand the concept and move on to the next step.

Q1. Define a function named my_greet that prints "Welcome." and call this function twice to print this greeting twice.

Condition for Execution	Welcome. Welcome.
Time	5 minutes

Write the entire code and the expected output results in the note.

Q2. Implement the max2(m, n) function, which takes two parameters named m and n, and returns the larger of these two values, and the min2(m, n) which also takes two parameters named m and n and returns the smaller of these two values. Assign 100 and 200 as arguments and call two functions to check the results.

Condition for Execution	The greater of 100 or 200 is : 200 The smaller of 100 or 200 is : 100
time	10 minutes

Write the entire code and the expected output results in the note.

Q3. We want to change the value of the mile, the unit mainly used in the United States, to the value of the kilometer, the international standard unit. Implement the mile2km(mi) function that takes a mile value as a parameter and returns it in kilometers and calls this function to output 1 to 5 miles as kilometers. In this case, use for - in range to make it repeatable. (Define 1 mile as 1.61 km.)

Condition for Execution	1 mile = 1.61 kilometers 2 mile = 3.22 kilometers 3 mile = 4.83 kilometers 4 mile = 6.44 kilometers 5 mile = 8.05 kilometers
Time	5 minutes

Write the entire code and the expected output results in the note.

Q4. Implement the cel2fah(cel) function that takes a temperature in Celsius (Celsius) as a parameter and returns it in Fahrenheit. Then, call this function to change from 10 to 50 degrees Celsius in units of 10 degrees, and output it in Fahrenheit temperature as the following result.

Condition for Execution	10 degrees Celsius = 50.0 degrees Fahrenheit 20 degrees Celsius = 68.0 degrees Fahrenheit 30 degrees Celsius = 86.0 degrees Fahrenheit 40 degrees Celsius = 104.0 degrees Fahrenheit 50 degrees Celsius = 122.0 degrees Fahrenheit
Time	5 minutes

conversion formula : Fahrenheit = Celsius × 9/5 + 32

Write the entire code and the expected output results in the note.

Pair programming

Pair Programming Practice

Guideline, mechanisms & contingency plan

Preparing pair programming involves establishing guidelines and mechanisms to help students pair properly and to keep them paired. For example, students should take turns “driving the mouse.” Effective preparation requires contingency plans in case one partner is absent or decides not to participate for one reason or another. In these cases, it is important to make it clear that the active student will not be punished because the pairing did not work well.

Pairing similar, not necessarily equal, abilities as partners

Pair programming can be effective when students of similar, though not necessarily equal, abilities are paired as partners. Pairing mismatched students often can lead to unbalanced participation. Teachers must emphasize that pair programming is not a “divide-and-conquer” strategy, but rather a true collaborative effort in every endeavor for the entire project. Teachers should avoid pairing very weak students with very strong students.

Motivate students by offering extra incentives

Offering extra incentives can help motivate students to pair, especially with advanced students. Some teachers have found it helpful to require students to pair for only one or two assignments.

Pair Programming Practice

Prevent collaboration cheating

The challenge for the teacher is to find ways to assess individual outcomes, while leveraging the benefits of collaboration. How do you know whether a student learned or cheated? Experts recommend revisiting course design and assessment, as well as explicitly and concretely discussing with the students on behaviors that will be interpreted as cheating. Experts encourage teachers to make assignments meaningful to students and to explain the value of what students will learn by completing them.

Collaborative learning environment

A collaborative learning environment occurs anytime an instructor requires students to work together on learning activities. Collaborative learning environments can involve both formal and informal activities and may or may not include direct assessment. For example, pairs of students work on programming assignments; small groups of students discuss possible answers to a professor’s question during lecture; and students work together outside of class to learn new concepts. Collaborative learning is distinct from projects where students “divide and conquer.” When students divide the work, each is responsible for only part of the problem solving and there are very limited opportunities for working through problems with others. In collaborative environments, students are engaged in intellectual talk with each other.

Q1. Let the user input three integers a, b, and c. And print the average, maximum, and minimum values of these three numbers as follows. In this case, mean3(a, b, c), max3(a, b, c), min3(a, b, c) that takes three numbers as parameters and returns the average, maximum, and minimum values of these three numbers. Define and call each function.

Output Example

```
Enter three numbers : 9 2 6
The average value of 9, 2, 6 is 5.666666666666667
The maximum value of 9, 2, 6 is 9
The minimum value of 9, 2, 6 is 2
```

Paper coding

Try to fully understand the basic concept before moving on to the next step.

Lack of understanding basic concepts will increase your burden in learning this course, which may make you fail the course.

It may be difficult now, but for successful completion of this course we suggest you to fully understand the concept and move on to the next step.

Q1. Let's take a number n as input and find the sum from 1 to n. Write this function using a recursive function call.

Condition for Execution	Enter a number : 10 55
Time	5 minutes

Write the entire code and the expected output results in the note.

Q2. Python has the ** operator, which indicates a square. However, let's take x and n as inputs without using an operator and use a recursive function to output x to the nth power. Let's try to output 2^{10} by inputting 2 as the x value and 10 as the n value as follows.

Condition for Execution	Enter x : 2 Enter n : 10 1024
Time	5 minutes

Write the entire code and the expected output results in the note.

Pair Programming Practice

Guideline, mechanisms & contingency plan

Preparing pair programming involves establishing guidelines and mechanisms to help students pair properly and to keep them paired. For example, students should take turns “driving the mouse.” Effective preparation requires contingency plans in case one partner is absent or decides not to participate for one reason or another. In these cases, it is important to make it clear that the active student will not be punished because the pairing did not work well.

Pairing similar, not necessarily equal, abilities as partners

Pair programming can be effective when students of similar, though not necessarily equal, abilities are paired as partners. Pairing mismatched students often can lead to unbalanced participation. Teachers must emphasize that pair programming is not a “divide-and-conquer” strategy, but rather a true collaborative effort in every endeavor for the entire project. Teachers should avoid pairing very weak students with very strong students.

Motivate students by offering extra incentives

Offering extra incentives can help motivate students to pair, especially with advanced students. Some teachers have found it helpful to require students to pair for only one or two assignments.

Pair Programming Practice

Prevent collaboration cheating

The challenge for the teacher is to find ways to assess individual outcomes, while leveraging the benefits of collaboration. How do you know whether a student learned or cheated? Experts recommend revisiting course design and assessment, as well as explicitly and concretely discussing with the students on behaviors that will be interpreted as cheating. Experts encourage teachers to make assignments meaningful to students and to explain the value of what students will learn by completing them.

Collaborative learning environment

A collaborative learning environment occurs anytime an instructor requires students to work together on learning activities. Collaborative learning environments can involve both formal and informal activities and may or may not include direct assessment. For example, pairs of students work on programming assignments; small groups of students discuss possible answers to a professor’s question during lecture; and students work together outside of class to learn new concepts. Collaborative learning is distinct from projects where students “divide and conquer.” When students divide the work, each is responsible for only part of the problem solving and there are very limited opportunities for working through problems with others. In collaborative environments, students are engaged in intellectual talk with each other.

Q1. The natural number e, also called Euler's number or Napier's constant, is an irrational number used as the base of the natural logarithm. It is defined by the following formula.

(e = 1 + 1/1! + 1/2! + 1/3! + ... + 1/n!)

In this formula, let k! be defined as a function named factorial(k). Also, let's define a function called euler(n) that returns the sum of 1/0! + 1/1! to 1/n! Find the value of euler(20) to five decimal places and output it as follows. (You must use a recursive function.)

Output example

eular(20) = 2.71828

Paper coding

Try to fully understand the basic concept before moving on to the next step.

Lack of understanding basic concepts will increase your burden in learning this course, which may make you fail the course.

It may be difficult now, but for successful completion of this course we suggest you to fully understand the concept and move on to the next step.

Q1

There is a list with integer element values called n_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]. Return even_list which only contains items of even number values from n_list by using the filter function and the lambda function.

- The returned even_list = [2, 4, 6, 8, 10]

Conditions for Execution	even_list = [2, 4, 6, 8, 10]
Time	5 Minutes

Create an empty list named even_list and add even value items by the append method. Use the for statement and the filter function. Use lambda function inside the filter function.

Q2. There is a list with integer unit values called n_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]. Return even_list which only contains items of even number values from n_list by using a lambda function. This time, do not use for statement, and instead use list function.

Returned even_list = [2, 4, 6, 8, 10]

Conditions for Execution	even_list = [2, 4, 6, 8, 10]
Time	5 Minutes

Modify objects returned by the filter function to list objects by the list function, and assign them to even_list.

Write the entire code and the expected output results in the note.

Write the entire code and the expected output results in the note.

Q3. Write a map function that converts a_list which contains lowercase alphabets like [‘a’, ‘b’, ‘c’, ‘d’] to a upper_a_list which contains upper case alphabets like [‘A’, ‘B’, ‘C’, ‘D’].

Also, define a function named to_upper that receives lowercase letters as parameters and returns uppercase letters, and convert those lowercase letters.

Conditions for Execution	`upper_a_list = ['A', 'B', 'C', 'D']`
Time	5 Minutes

Write the entire code and expected output result.

Write the entire code and the expected output results in the note.

Q4. Compute the sum of integers from 1 to 100 by using reduce function and lambda expression inside it. Use range (1, 101) as an input.

Condition for Execution	Sum of 1 to 100 : 5050
Time	5 Minutes

Pair programming

Pair Programming Practice

Guideline, mechanisms & contingency plan

Preparing pair programming involves establishing guidelines and mechanisms to help students pair properly and to keep them paired. For example, students should take turns “driving the mouse.” Effective preparation requires contingency plans in case one partner is absent or decides not to participate for one reason or another. In these cases, it is important to make it clear that the active student will not be punished because the pairing did not work well.

Pairing similar, not necessarily equal, abilities as partners

Pair programming can be effective when students of similar, though not necessarily equal, abilities are paired as partners. Pairing mismatched students often can lead to unbalanced participation. Teachers must emphasize that pair programming is not a “divide-and-conquer” strategy, but rather a true collaborative effort in every endeavor for the entire project. Teachers should avoid pairing very weak students with very strong students.

Motivate students by offering extra incentives

Offering extra incentives can help motivate students to pair, especially with advanced students. Some teachers have found it helpful to require students to pair for only one or two assignments.

Pair Programming Practice

Prevent collaboration cheating

The challenge for the teacher is to find ways to assess individual outcomes, while leveraging the benefits of collaboration. How do you know whether a student learned or cheated? Experts recommend revisiting course design and assessment, as well as explicitly and concretely discussing with the students on behaviors that will be interpreted as cheating. Experts encourage teachers to make assignments meaningful to students and to explain the value of what students will learn by completing them.

Collaborative learning environment

A collaborative learning environment occurs anytime an instructor requires students to work together on learning activities. Collaborative learning environments can involve both formal and informal activities and may or may not include direct assessment. For example, pairs of students work on programming assignments; small groups of students discuss possible answers to a professor's question during lecture; and students work together outside of class to learn new concepts. Collaborative learning is distinct from projects where students "divide and conquer." When students divide the work, each is responsible for only part of the problem solving and there are very limited opportunities for working through problems with others. In collaborative environments, students are engaged in intellectual talk with each other.

Q1. If you treat students' scores of English, math, and science exams as list of three elements, it can be expressed as a list such as [100, 90, 95]. If there are two students, their scores can be expressed as [100, 90, 95, 90, 85, 93]. If a student did not apply to the exam of a certain subject, denote that score as 0. Print how many students' scores are contained in the given scores list, the number of students with valid scores for all subjects (that is students with no 0 for all subject), and the scores of students with only valid scores.

Example of Input

```
scores = [100, 90, 95, 90, 80, 70, 0, 80, 90, 90, 0, 90, 100, 75, 20, 30, 50, 90]
```

Example of

```
scores = [100, 90, 95, 90, 80, 70, 0, 80, 90, 90, 0, 90, 100, 75, 20, 30, 50, 90]
The number of total students is 6.
The number of students with valid scores is 4.
[[100, 90, 95], [90, 80, 70], [100, 75, 20], [30, 50, 90]]
```

Write the entire code and the expected output results in the note.

Paper coding

Try to fully understand the basic concept before moving on to the next step.

Lack of understanding basic concepts will increase your burden in learning this course, which may make you fail the course.

It may be difficult now, but for successful completion of this course we suggest you to fully understand the concept and move on to the next step.

Q1. Create a nested-function by defining a function named greetings and another function named say_hi inside that function. Call say_hi function within greetings. Then call greetings and print ‘hello’. say_hi function is shown below.

```
def  say_hi():
        print(‘hello’)
```

Condition for Execution	`hello`
Time	5 Minutes

Write the entire code and the expected output results in the note.

Q2. Write the following function calc and assign calc to variable num. Then, execute num(3). Make the execution result 14 as follows.

```
def calc():
    a = 3
    b = 5
    def mul_add(x):
        return a * x + b
    return mul_add
```

Condition for Execution	14
Time	5 Minutes

Write the entire code and the expected output results in the note.

Q3. Build mul_add, the inner function of the nested function calc from the previous problem, by using lambda expressions, and print the following result.

Condition for Execution	14
Time	5 Minutes

Write the entire code and the expected output results in the note.

Pair programming

Pair Programming Practice

Guideline, mechanisms & contingency plan

Preparing pair programming involves establishing guidelines and mechanisms to help students pair properly and to keep them paired. For example, students should take turns “driving the mouse.” Effective preparation requires contingency plans in case one partner is absent or decides not to participate for one reason or another. In these cases, it is important to make it clear that the active student will not be punished because the pairing did not work well.

Pairing similar, not necessarily equal, abilities as partners

Pair programming can be effective when students of similar, though not necessarily equal, abilities are paired as partners. Pairing mismatched students often can lead to unbalanced participation. Teachers must emphasize that pair programming is not a “divide-and-conquer” strategy, but rather a true collaborative effort in every endeavor for the entire project. Teachers should avoid pairing very weak students with very strong students.

Motivate students by offering extra incentives

Offering extra incentives can help motivate students to pair, especially with advanced students. Some teachers have found it helpful to require students to pair for only one or two assignments.

Pair Programming Practice

Prevent collaboration cheating

The challenge for the teacher is to find ways to assess individual outcomes, while leveraging the benefits of collaboration. How do you know whether a student learned or cheated? Experts recommend revisiting course design and assessment, as well as explicitly and concretely discussing with the students on behaviors that will be interpreted as cheating. Experts encourage teachers to make assignments meaningful to students and to explain the value of what students will learn by completing them.

Collaborative learning environment

A collaborative learning environment occurs anytime an instructor requires students to work together on learning activities. Collaborative learning environments can involve both formal and informal activities and may or may not include direct assessment. For example, pairs of students work on programming assignments; small groups of students discuss possible answers to a professor’s question during lecture; and students work together outside of class to learn new concepts. Collaborative learning is distinct from projects where students “divide and conquer.” When students divide the work, each is responsible for only part of the problem solving and there are very limited opportunities for working through problems with others. In collaborative environments, students are engaged in intellectual talk with each other.

Q1. Extract list result from list lst which has values of 1 ~ 100. list result has elements of list lst that are divisible by 5 or 7. Declare func1(a) function and nest func2 and func3 function. Then, call the two functions in func1 function and print numbers divisible by 5 or 7. Here, align the values by using the sorted () function.

Print example

```
def func2():
    result1 = []
    for i in a:
        if i % 5 == 0:
            result1.append(i)
    return result1

def func3():
    result2 = []
    for i in a:
        if i % 7 == 0:
            result2.append(i)
    return result2
```

```
lst =  [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31,
32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 6
2, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92,
93, 94, 95, 96, 97, 98, 99, 100]
result = [5, 7, 10, 14, 15, 20, 21, 25, 28, 30, 35, 35, 40, 42, 45, 49, 50, 55, 56, 60, 63, 65, 70, 70, 75, 77, 80, 84, 8
5, 90, 91, 95, 98, 100]
```

Paper coding

Try to fully understand the basic concept before moving on to the next step.

Lack of understanding basic concepts will increase your burden in learning this course, which may make you fail the course.

It may be difficult now, but for successful completion of this course we suggest you to fully understand the concept and move on to the next step.

Q1 Construct class Dog and its objects with the functionalities described below.

- , a method named def bark(self): . This method prints a barking sound.

b) Generates an instance named Dog and refers my_dog by a command named my_dog=Dog.

c) Prints a barking sound with a method named my_dog.bark()

"woof woof"

Condition for Execution	woof woof
Time	5 Minutes

Write the entire code and the expected output results in the note.

Q2 Define class Dog with the functionalities described below and call instances and methods.

- a) This class Dog has an attribute named name.
  b) Has an initialize method named def __init__(self, name): . This method initializes Dog's name.
  c) Has a method named def bark(self). This method prints a barking sound.
  d) Generates a my_dog instance that has name 'Bingo' with the command my_dog=Dog('Bingo')
  e) Prints the following barking sound with the method my_dog.bark()
  "Bingo : woof woof"

Condition for Execution	Bingo : woof woof
Time	5 Minutes

Write the entire code and the expected output results in the note.

Pair Programming Practice

Guideline, mechanisms & contingency plan

Preparing pair programming involves establishing guidelines and mechanisms to help students pair properly and to keep them paired. For example, students should take turns “driving the mouse.” Effective preparation requires contingency plans in case one partner is absent or decides not to participate for one reason or another. In these cases, it is important to make it clear that the active student will not be punished because the pairing did not work well.

Pairing similar, not necessarily equal, abilities as partners

Pair programming can be effective when students of similar, though not necessarily equal, abilities are paired as partners. Pairing mismatched students often can lead to unbalanced participation. Teachers must emphasize that pair programming is not a “divide-and-conquer” strategy, but rather a true collaborative effort in every endeavor for the entire project. Teachers should avoid pairing very weak students with very strong students.

Motivate students by offering extra incentives

Offering extra incentives can help motivate students to pair, especially with advanced students. Some teachers have found it helpful to require students to pair for only one or two assignments.

Pair Programming Practice

Prevent collaboration cheating

The challenge for the teacher is to find ways to assess individual outcomes, while leveraging the benefits of collaboration. How do you know whether a student learned or cheated? Experts recommend revisiting course design and assessment, as well as explicitly and concretely discussing with the students on behaviors that will be interpreted as cheating. Experts encourage teachers to make assignments meaningful to students and to explain the value of what students will learn by completing them.

Collaborative learning environment

A collaborative learning environment occurs anytime an instructor requires students to work together on learning activities. Collaborative learning environments can involve both formal and informal activities and may or may not include direct assessment. For example, pairs of students work on programming assignments; small groups of students discuss possible answers to a professor’s question during lecture; and students work together outside of class to learn new concepts. Collaborative learning is distinct from projects where students “divide and conquer.” When students divide the work, each is responsible for only part of the problem solving and there are very limited opportunities for working through problems with others. In collaborative environments, students are engaged in intellectual talk with each other.

Q1

Construct class Student that has following functionalities.
This student took quizzes for English, mathematics and science, and the quiz scores are given as inputs.
Generate instances by using this class. This class has the following attributes and actions.

attributes

name : student name stored in string type.
student_id : student ID such as s2020001 stored in 8-digit integers.
eng_quiz : student's English quiz core stored in list format.
math_quiz : student's math quiz score stored in list format.
science_quiz : student's science quiz score stored in list format.

actions(methods)

__init__ : initializes with the student's name and studentID.
__str__ : returns the student's name, studentID and quiz scores as strings

set_eng_quiz : sets the student's English quiz.
set_math_quiz : sets the student's mathematics quiz.
set_science_quiz : sets the student's science quiz.

get_name : returns the student's name
get_student_id : returns the studentID
get_eng_quiz : returns the student's English quiz
get_math_quiz : returns the student's math quiz
get_science_quiz : returns the student's science quiz
get_total_score : returns the student's total quiz score
get_avg_score : returns the student's average quiz score

Q1

Construct class Student that has following functionalities.
This student took quizzes for English, mathematics and science, and the quiz scores are given as inputs.
Generate instances by using this class. This class has the following attributes and actions.

Output example

```
Enter the student's name : David Doe
Enter the student's ID : 20213093
Enter the student's English quiz score : 90
Enter the student's mathematics quiz score : 95
Enter the student's science quiz score : 100
Name : David Doe, ID : 20213093
English quiz score : 90, Mathematics quiz score : 95
Science quiz score : 100,
Total : 285, Average : 95.0
```