1. Consider the modular structure of a program to create posters in three stages: painting background colors, inserting pictures, and printing letters. For each of these three stages we have in and out functions. The in function takes instructions for each stage, while the out function renders the design on paper.

We can design the program in two ways:

(a) There are two modules: input and output. The input module collects all in functions for each stage. All out functions are contained in the module output.

(b) There is a module for each stage. Each module contains the specific in and out functions for the stage.

Which design would be best? Justify using the principles of good modular design.

**Answer:**

The second design works best.

Justification along the three principles of good modular design:

(a) information hiding: Whoever writes the modules in the first design needs to know the ins or outs of all stages in the creation of a poster. A programmer in the second design can focus on one stage, each module hides details about its stage to the other modules.

(b) high cohesion and low coupling: There is high cohesion in the second design, because all the functionality of one stage is concentrated in one module, while the functionality of one stage is divided among several modules in the first design. There is high coupling in the first design about the in and out functions for the same stage. In the second design the programmer of one module can use common code and conventions for the in and out functions for each stage.

(c) design for change: If another stage is added to the creation of the poster, in the second design we have only to add another module, the existing modules for the other stages stay the same. In the first design, we must change all existing modules.
2. What does “to copyleft software” mean? Who can do this? What are the consequences of this?

**Answer:**

To copyleft software means to reverse the copyright on the software. Only the one who holds the copyrights on the software can do this. As a consequence, the original copyright holder shares the rights on the software with everyone else.

3. The memory consumption of an algorithm grows like $O(n^2)$ where $n$ is the dimension of the input. Suppose the algorithm requires 0.5 Gigabyte of memory to process inputs of dimension $n = 1000$. If we have only two Gigabytes of memory available, how large can $n$ be? Justify your answer.

**Answer:** $n = 2000$

Justification: $O(n^2)$ means that as we double the input size $n$, the cost is multiplied by 4, and $4 \times 0.5$ Gigabytes = 2 Gigabytes.

4. Give Python code to define the following function:

```python
def GetValue():
    """
    Asks the user for a floating-point number. As long as the user input cannot be converted into a float, an error message is printed and the user is invited to try again. The function returns the float given by the user.
    """

    while True:
        a = raw_input('give a floating-point number : ')
        try:
            x = float(a)
            return x
        except:
            print 'wrong input, please try again'
```

**Answer:**
5. Write a program that prompts the user for the name of a Python script.
   The program counts all the print statements and appends all arguments of each print
   to a list. After reading the entire script, the program writes the number of print
   statements and the list of all arguments that occurred in a print.
   The length of the list must equal the number of print statements, so arguments of one
   print separated by spaces must be joined. You may assume the script has at most one
   statement per line, but do not make any assumptions about spaces or tabs.

   **Answer:**

   ```python
   name = raw_input('give the name of the script : ')
   file = open(name,'r')
   P = []
   cnt = 0
   while True:
       s = file.readline()
       if s == '': break
       L = s.split(' ')
       if 'print' in L:
           cnt = cnt + 1
           i = L.index('print')
           a = ''.join(L[i+1:])
           P.append(a)
   print '#prints : ', cnt, 'arguments : ', P
   file.close()
   ```
6. Draw a use case diagram to distinguish the actions on an object of the class car which can be performed by two different types of users: passenger and driver. Indicate also those object data attributes of a car, most relevant to the distinction between its users.

**Answer:**

![Use Case Diagram]

A passenger may not invoke the drive method of the class Car. In order to drive, the driver has to sit in the front seat, while passengers can only occupy back seats. Both passenger and driver can step out the car. So **front_seat** and **back_seats** are the data attributes used to distinguish drivers from passengers.

7. A circle is determined by three parameters: the two coordinates $x$ and $y$ of its center, and its radius $r$. Draw the design of a GUI to visualize circles for various values of the three parameters $x$, $y$, and $r$. Label each component of the GUI with the proper name of the widget. Do not give any Python code.

**Answer:**

![GUI for Circle Visualization]

We have used four labels, one label to indicate the working of the GUI and the other three labels for the scales. Three scales are used to enter the three parameters. Finally, the canvas shows the circle.