|  | United International University (UIU)Department of CSETrimester: Spring 2022Course Name: | CSI 424 | Simulation & Modeling Laboratory (Section A) |
| --- | --- |

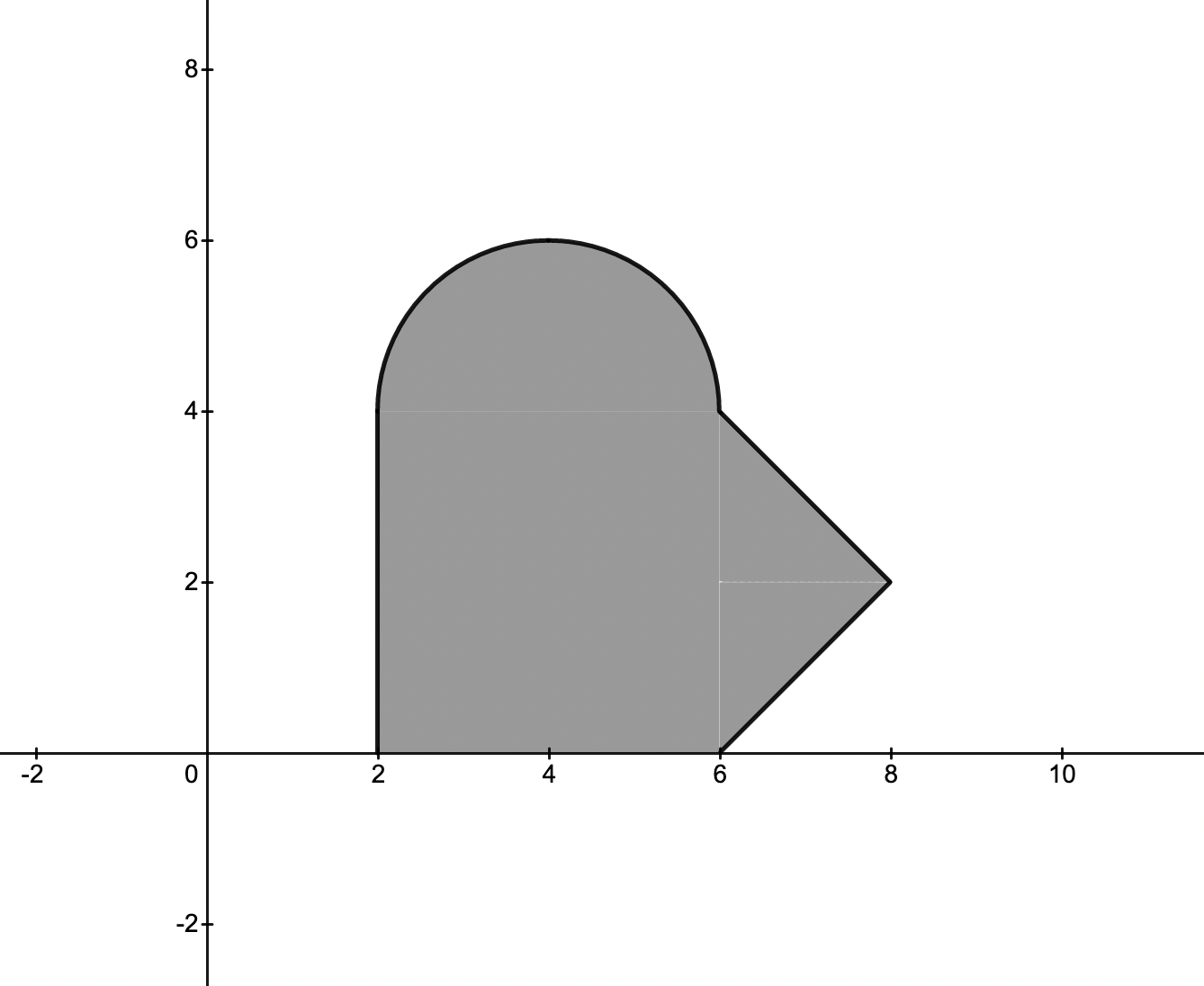
## Submission Guideline:

* Please solve the problems in separate files (**One notebook/python file per task**).
* **Download the python files** as instructed in the class. (File -> Download -> Download .py)
* Create a new **folder** and put all your python files inside the folder.
* Rename the folder with your 9 digit student ID.
* Make a ZIP of the folder and **submit the .zip file**.

Please do not copy codes from others/the internet. Each of the offline assignments will be evaluated with a viva. You must be able to explain your code. Also, we will run a copy checker on the submissions. Any plagiarism will be severely penalized.

**Offline assignment 2**

1. **[4 marks]** Use Monte Carlo method to **estimate the area** of the shaded region. Also make a **scatter plot** of your sample points as we did in our lecture.



1. **[2 marks]** Use Monte Carlo integration to **estimate the integral** of a function. Also **calculate the error**.

You have to solve only one of the following integration problems. The problem you need to solve is determined by the last digit of your student ID. For example: if your student ID is *011123456*, you will solve problem-6.

0.

2. **[4 marks]** Simulate Conway's game of life with a binary matrix of size (12,12) for 10 timestamps. Print the 2d array at each timestamp.

We will use 1 to denote an alive cell and 0 to denote a dead cell.

There are four simple rules in the Game of Life.

* Each living cell with one or no neighbors alive dies, as if by solitude.
* Each living cell with four or more neighbors alive dies, as if by overpopulation.
* Each living cell with two or three neighbors alive survives.
* Each dead cell with three neighbors alive becomes populated.

| Initial grid (t=0) | t=1 | t=2 |
| --- | --- | --- |
| 000000000000  000000000000  000000000000  000000000000  00000**1**000000  00000**11**00000  00000**1**000000  000000000000  000000000000  000000000000  000000000000  000000000000 | 000000000000  000000000000  000000000000  000000000000  00000**11**00000  0000**111**00000  00000**11**00000  000000000000  000000000000  000000000000  000000000000  000000000000 | 000000000000  000000000000  000000000000  000000000000  0000**1**0**1**00000  0000**1**00**1**0000  0000**1**0**1**00000  000000000000  000000000000  000000000000  000000000000  000000000000 |

Generate results upto t=10.