Лабораторная 1

Рекурсия. Ханойская башня. Написать рекурсивную функцию для решения задачи о Ханойских башнях. Найти и доказать необходимое и достаточное количество шагов для решения задачи.

Лабораторная 2

Программа печати книги. Есть устройство - принтер. Принтер может печатать только с одной стороны. У принтера есть лоток для подачи бумаги и лоток выдачи. Если пронумеровать листы в лотке, начиная с верхнего, и запустить печать на `n` листов, то по окончании печати в лотке выдачи свежая печать будет на листах с нижней стороны, первый лист окажется в самом низу, а последний в самом верху. Перед нами стоит задача: распечатать книгу, размещая с каждой стороны листа по две страницы:

```
 _____________      _____________
| ==== | ==== |    | ==== | ==== |
| ==== | ==== |    | ==== | ==== |
| ==== | ==== |    | ==== | ==== |
|2_____|____11|    |4_____|_____9|
```

Затем стопка из лотка выдачи перемещается снова в лоток подачи, и печать повторяется для другой стороны (только надо учесть, что порядок листов стал обратным). Распечатанные листы затем сшиваются вместе по центральной вертикальной линии, образующей сгиб. В результате получаем брошюру, в которой должны быть правильно пронумерованы страницы. Много страниц в одну брошюру не сошьешь, поэтому большую книгу делят на несколько частей, каждую из которых печатают и сшивают. Затем все части склеивают в единую книгу. Необходимо написать программу, которая формирует программу печати. Программа печати – это список элементов печати. Элемент печати - это одна из следующих сущностей:

- Пара чисел – номера двух страниц, которые надо напечатать на текущей стороне листа. Если печатать страницу не надо, то вместо номера надо указать 0. Пример: `(1, 10)`, `(0, 8)`
- Признак смены стороны. Служит, чтобы сообщить пользователю о завершении печати на одной стороне листов и о необходимости переместить стопку листов из локта выдачи в лоток подачи. Пример: `@`
- Признак смены части. Служит, чтобы сообщить пользователю о завершении печати части книги на обоих сторонах стопки листов и необходимости загрузки новой стопки чистых листов. Пример: `@@` На вход программа получает 2 числа: количество страниц подготовленных к печати и количество листов в каждой "брошюре".

В качестве результата программа должна вывести на экран программу печати. Пример для 15 страниц и 3 листов в "брошюре":

```
[(12, 1), (10, 3), (8, 5), `@`, (6, 7), (4, 9), (2, 11), `@@`, (0, 13), (0,
15), (0, 0), `@`, (0, 0), (0, 0), (14, 0)]
```

Использовать подход TDD: сначала пишем тесты, потом функциональный код.

Лабораторная 3

Начальные этапы разработки ПО. Написать список требований, архитектуру и решения, которые были приняты для обеспечения атрибутов качества для программного продукта. Что за продукт, обсуждалось на занятиях.

Лабораторная 4

Написать программы, которые, используя регулярные выражения, из данной строки (а желательно из файла) отыскивают следующие регулярные конструкции:

1. Корректные IP-адреса вида xxx.xxx.xxx.xxx.
   Например: `0.0.0.0`, `192.168.0.1`, `195.19.32.101`. В то же время не отыскивать следующие конструкции (и их части): `1.14.1.01`, `1.2.3.4.5`, `254.255.256.257`, `1111.1.1.1.1111`.
2. Qualified identifiers. Названия переменных и функций, в том числе и тех, к которым обращаются косвенно.
   Например: `value`, `a1`, `i`, `math.sqrt`, `datetime.datetime.now`. В то же время не отыскивать следующие конструкции (и их части): `1.a`, `a.1b`, `a..b`, `.a`, `a.`. То есть, найти идентификаторы и цепочки идентификаторов, разделенных точкой. Уметь находить корректные Qualified identifiers не только в середине текста, но и в начале и в конце.
3. Время суток вида `HH:MM [PM/AM]`. Например: `23:55`, `1:30 PM`, `07:15AM`. В то же время не отыскивать следующие конструкции (и их части): `12:5`, `25:10`, `12:75`, `12:55:55`, `:40`, `12:`, `1:30 AMX`, `11:42 PM1`. То есть все времена как в 24-часовом формате, так и в 12-часовом. Если время может трактоваться как в 12ти так и в 24-часовом формате, выбирать тот, который имеет наибольшую длину (12-часовой), например в строке `Они решили собраться в 09:00 PM`. Хотя и `09:00`, и `09:00 PM` являются корректными временами суток, выбирается последний вариант.

Уметь находить все корректные вхождения строк: не только в середине текста, но и в начале и в конце.

Лабораторная 5

Имеется папка с большим количеством (10 000) файлов. В каждом из файлов описана фигура (треугольник или четырехугольник) в следующем формате:

```
figure=ACBD
```

```
B(3,4,5)
C(0,0,0)
A(0,1,2)
D(4.5, 1.7e+1, -1e-9)
```

Отнести фигуру к одному из следующих классов:

1. convex. Невырожденный несамопересекающийся выпуклый n-угольник.
2. concave. Невырожденный несамопересекающийся невыпуклый n-угольник.
3. other. Не отнесенный ни к одному из вышеперечисленных классов (самопересекающийся или вырожденный).

После классификации необходимо переместить (либо скопировать) файлы в субдиректории: `./convex`, `./concave`, `./other`, - соответствующие классу.

Написать параллельную версию программы (для Python использовать модуль `multiprocessing` и в нем класс `Pool`). Сравнить производительность работы классификатора для разных количеств процессов (1, 2, 4, 8, 16...). Найти оптимальное количество процессов для своей машины.