Домашнее задание 3

Если Вам не удалось установить SimpleGUICS2Pygame или pygame

Для выполнения этого и следующих заданий необходимо иметь установленные библиотеки SimpleGUICS2Pygame и pygame. Их установка описана в первом задании («Установите Python»). Если эти библиотеки установить не удалось, можно использовать онлайн версию Python, которая находится по адресу http://www.codeskulptor.org/.

Учтите, что там используется 2-я версия Python, поэтому, скорее всего не будет правильно работать вывод текста на русском языке, используйте английский язык.

Есть ещё два отличия, от используемой на занятиях версии:

1. В CodeSculptor для подключения библиотеки SimpleGUI надо в начале программы вместо

```
import SimpleGUICS2Pygame.simpleguics2pygame as simplegui
```

писать просто

```
import simplegui
```

2. Для печати в консоль вместо `print(a)` надо писать `print a` (без круглых скобок).

Задание - игра «угадай число»

«Угадай число» это одна из простейших игр для двух игроков. Один игрок задумывает число из известного диапазона, второй пытается угадать это число. После каждой попытки первый игрок отвечает «больше», «меньше» или «угадал», в зависимости от того, является задуманное им число большим, меньшим или равным предложенному. В этом задании Вам надо реализовать программу на Python, в которой компьютер будет загадывать числа, а пользователь – угадывать.

Взаимодействие с программой будет организовано с помощью поля ввода и нескольких кнопок. В этом задании не требуется использовать холст, результаты игры будут печататься в консоли[1].

Рекомендуемый процесс решения

В качестве основы используйте шаблон numbers_template.py. Рекомендуется решать задачу в следующем порядке.

1. Определите, какие глобальные переменные потребуются для хранения состояния игры. Например, достаточно очевидным представляется хранить число, «задуманное» (сгенерированное случайным образом) программой. Потребуются и другие переменные, особенно для реализации дополнительных расширений.

[1] Сначала реализовать программу так, чтобы она печатала данные в консоли, а уже потом реализовать графический интерфейс – хорошая тактика и для задач в дальнейшем. Это позволяет сначала сфокусироваться на логике программы и правильно её реализовать, а уже затем задумываться о том, как её красиво представить. Поиск ошибок в программе, когда она работает в режиме графического интерфейса часто оказывается сложным.

2. Вспомните, как в Python сгенерировать случайное число в заданном диапазоне от `low` до `high`. При обсуждении диапазонов будем следовать принятому в Python соглашению, что числа диапазона включают нижнюю границу, и не включают верхнюю, что в математической записи выражается как `[low,high)`. Таким образом, `[0,3)` включает числа `0`, `1` и `2`. Начните свою реализацию с работы с диапазона `[0,100)`.
3. Вспомните, как создать поле ввода текста, используя модуль simplegui. Это поле будет использоваться для ввода предположения пользователя.
4. Напишите функцию-обработчик `input_guess(guess)`, которая принимает предположение (англ. gues) пользователя, сравнивает с задуманным числом и печатает соответствующее сообщение.
   Подсказка: потребуется преобразовать предположение пользователя из строки в число.
5. Напишите код для создания окна, поля ввода и запуска окна.
6. Проверьте Ваш код запуская игру несколько раз. На этом этапе после завершения игры программу надо закрыть и запустить заново. Вы также можете использовать шаблон numbers_test.py для тестирования.
7. Напишите функцию new_game(), которая должна начинать новую игру. Внутрь этой функции поместите генерацию случайного числа. Поместите вызов этой функции перед созданием окна.

На этом этапе Вы получите минимальную по возможностям программу, которая может быть сдана. Дальнейшие доработки помогут Вам получить дополнительные баллы.

Дополнительные расширения игры

1. Добавьте две кнопки для перезапуска игры с разными диапазонами. Это должны быть кнопки «Диапазон: 0-100» и «Диапазон: 0-1000». Нажатие на любую из этих кнопок должно приводить к перезапуску игры и печати соответствующего сообщения. Они должны работать в любой момент времени во время игры.
   При запуске программы она должна сразу же начинать игру в диапазоне 0-100. Когда игра завершилась, она должна сразу же перезапуститься в том же диапазоне, в котором проходила.
   Подсказка: при реализации этого пункта может быть полезным завести глобальную переменную.

Играя в «угадай число», можно заметить, что хорошей стратегией является запоминание диапазона, состоящего из максимального предположения которое «меньше» чем задуманное число, и минимального, которое «больше». Хорошим кандидатом на следующее предположение будет среднее арифметическое между этими двумя числами. Ответ на это предположение позволит определить следующий интервал, содержащий задуманное число, и он будет в два раза меньше исходного. Например, если игра проходит в диапазоне [0,100) можно начать с предположения 50. Если ответом будет «больше», значит задумано число в диапазоне [51,100). Далее можно предложить 75, и так далее. Такой способ сокращения пространства поиска в программировании известен как двоичный поиск.

Следуя идеи двоичного поиска можно угадать любое число в диапазоне $[low, high)$ не более чем за n шагов, где n – минимальное число, такое что $2^n \geq high - low + 1$. Для диапазона $[0{,}100)$ $n = 7$, для диапазона $[0{,}1000)$ $n = 10$.

2. Вторым добавлением к «угадай число» будет ограничение числа попыток, за которые игрок должен угадать число. После каждой попытки игра должна печатать число оставшихся попыток. Если игрок использовал все попытки, то он проигрывает, и игра должна распечатать соответствующее сообщение.
   В качестве ограничений, используйте числа, указанные выше. Подумайте, как бы можно было вычислить соответствующее число по заданным границам диапазона, используя функции `math.log()` и `math.ceil()`.

Оценка задания.

Игра без дополнительных расширений	-	60%
Реализация кнопок для выбора диапазона	-	20%
Реализация ограничения на число попыток	-	20%