Wstęp do programowania 2014
Pracownia 7

Uwaga: Na tej liście też będą wprawki (może ze słownikiem, albo z listami składanymi?). Podczas tych zajęć można oddawać zadania z listy piątej za 0.5 i późniejszych za 1.

Premia za tę listę wynosi 0.5, wlicza się do maksimum, przyznawana jest osobom, które zdobyły co najmniej 1.5p za zadania z tej listy.

Zadanie 1.(1pkt) Jesteś zatrudniony jako programista w zespole tworzącym rubrykę Słowne Rekreacje w pewnym czasopiśmie. Twój szef stwierdził, że w następnym numerze powinno znaleźć się dużo zagadek, które polegają na takim przestawianiu liter w parze wyrazów, żeby otrzymać imię i nazwisko jakiejś (sławnej) osoby. Oczywiście najlepiej, gdyby miały one przynajmniej cień sensu i były poprawne gramatyczne, ale nie jest to warunkiem absolutnie koniecznym. Przykładowe zagadki:

wsparł busole – Bolesław Prus
połknij okrakiem – Mikołaj Kopernik
słał wieszczom – Czesław Miłosz
cenne wmieszał – ??
ekspansja rzodkwi (ew. pojesz kwadransik)– ??
obca makabra (ew. baba mocarka)[1]– ??

Nie tylko rozwiązywanie, ale również wymyślanie takich zagadek okazuje się całkiem trudne. Twoim zadaniem jest dostarczenie narzędzia, które wspomoże w tym zakresie pracę redakcji. A konkretnie, powinieneś napisać program, który dla zadanego imienia i nazwiska (tekstu wejściowego) wypisuje wszystkie pary wyrazów, które w sumie są układalne z tekstu wejściowego i to w taki sposób, że wykorzystane są wszystkie literki (czyli są one permutacją tekstu wejściowego). Innymi słowy Twój program ma wypisywać potencjalne zagadki. Każda para powinna być wypisana tylko raz (czyli albo *obca makabra* albo *makabra obca*, ale nie obie). Przetestuj program na wybranych imionach i nazwiskach (w szczególności swoim własnym). Przedstaw prowadzącemu działanie programu na jednym imieniu i nazwisku (najlepiej takim, które daje wg Ciebie najzabawniejsze efekty).

Zadanie 2.(1pkt) Obraz w pliku tekstowym będziemy zapisywać w następujący sposób:

```
(255,0,0) (255,0,0) (255,0,0)
(255,0,255) (255,255,0) (255,0,0)
(0.0,0,255) (0.0,255,0) (255,0,0)
```

Każdy wiersz jest wierszem obrazu, opisy pikseli – trójki liczb RGB – nie zawierają spacji. Napisz program, który wczytuje obraz z pliku i wykorzystując moduł turtle rysuje go na ekranie (pikselami powinny być kwadraty, o zadanym, niezbyt dużym boku). Wykorzystaj funkcje `setx` oraz `sety`. Uwaga: jeżeli irytuje cię moduł turtle, możesz do tego zadania wykorzystać umieszczony na stronie wykładu moduł `pygame_array` (dokonując być może drobnych poprawek w kodzie).

Zadanie 3.(1pkt) *Gra w życie* toczy się na prostokątnej planszy podzielonej na kwadraty. W danym momencie pojedynczy kwadrat może być zajęty przez komórkę lub pusty. Stan planszy zmienia się w kolejnych etapach gry, przy czym to, czy dana komórka będzie zajęta lub pusta w etapie $n+1$ zależy od tego, czy była ona zajęta w etapie n, jak również od tego, jaka była zajętość sąsiadów tej komórki w etapie n. Obowiązują następujące zasady:

(i) Każda komórka ma 8 sąsiadów (czterech po bokach i czterech stykających się z nią rogami)

(ii) Plansza jest „zawinięta", to znaczy że pierwszy wiersz sąsiaduje z ostatnim, analogicznie pierwsza kolumna sąsiaduje z ostatnią.

[1] Nie należy próbować rekonstruować z powyższych zagadek poglądów politycznych czy artystycznych autora listy. Zagadki są takie, jakie się dało ułożyć z kilku wybranych nazwisk. A nazwiska z kolei wybierane są tak, by dało się ułożyć jakąś zagadkę.

(iii) Komórka, która nie ma sąsiadów (w etapie n) umiera z samotności (w etapie $n+1$), podobnie taka, która ma tylko jednego sąsiada

(iv) Komórka, która ma dwóch lub trzech sąsiadów przeżywa

(v) Komórka, która ma czterech lub więcej sąsiadów umiera (bo ją tłok strasznie męczy)

(vi) W polu, które w etapie n jest puste i ma dokładnie trzech sąsiadów (tzn. trzy sąsiednie pola są pełne), w etapie $n+1$ powstaje nowa komórka.

Napisz program symulujący grę w życie. Stan początkowy populacji powinien być zadany wielowierszowym napisem, na przykład:

```
x = """
##........#####
...###.........
...#.#...######
...###.........
........##.....
##....###......
.#.............
"""
```

w którym kropki oznaczają puste komórki, natomiast krzyżyki – pełne. Symulacja powinna przebiegać tak, że w każdym etapie powinien być wypisany stan planszy (również jako wielolinijkowy napis), po czym wykonanie powinno zostać wstrzymane do momentu, w którym użytkownik wciśnie enter (na przykład przez wywołanie `raw_input()`), po wciśnięciu entera przechodzimy do kolejnego etapu. Symulacja powinna się zatrzymać w momencie, w którym użytkownik wpisze ustalone słowo, na przykład `koniec`. Możesz założyć, że napis oznaczający stan początkowy populacji jest prawidłowy (czyli każdy wiersz ma tę samą długość i składa się jedynie z kropek i krzyżyków). Stan początkowy może być umieszczony explicite w programie (w formie podstawienia, jak powyżej) lub wczytany z pliku.

Uwaga: zadanie będzie miało kontynuację na kolejnych listach

Zadanie 4. (0.5pkt)$\star$ Wykorzystaj modulik `pygame_array` do wizualizacji gry w życie (warto zobaczyć i uruchomić najpierw `pygame_array_test`). Jeżeli nie korzystasz z komputerów pracowianych, musisz wcześniej zainstalować bibliotekę pygame (`www.pygame.org`).

Zadanie 5.(0.5-1.0pkt)$\star$ Wykorzystaj modulik `pygame_array` do stworzenia jakiejś własnej, ciekawej animacji. To zadanie będzie miało o tydzień dłuższy termin niż pozostałe, ocena zależy od wrażenia, jakie wywrzesz na prowadzącym pracownie.