Projekt zaliczeniowy z Programowania - zaliczenie eksternistyczne

Termin nadsyłania rozwiązań na platformę Kampus2 - 18.04.2021

11 marca 2021

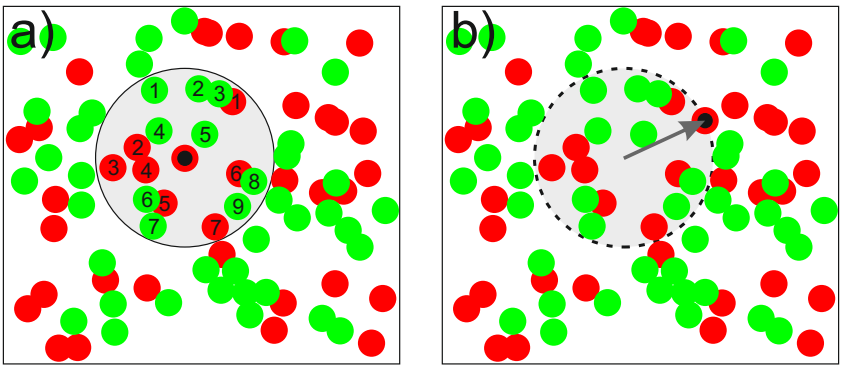
Program, który nie kompiluje się na komputerach w pracowni OKWF nie podlega ocenie.

Program powinien zostać napisany samodzielnie.

Prosimy o nadsyłanie rozwiązań na platformę Kampus2
`https://kampus-student2.ckc.uw.edu.pl/course/view.php?id=6442`
w postaci archiwum o nazwie *Imie_Nazwisko.zip* lub *Imie_Nazwisko.tar.gz*

Samodzielnicowanie się (samoorganizacja) miasta

Skrócony opis: Stwórzmy kwadratowe (na mapie) miasto o wielkości 25km × 25km – odpowiednik Warszawy. Miasto początkowo jest bezludne, a następnie zasiedlamy je **N** mieszkańcami, każdy w losowym miejscu w granicach miasta. Są dwa typy mieszkańców, dla rozróżnienia przypiszemy im kolory zielony i czerwony – np. kibice Legii i Polonii. Załóżmy, że połowa miasta kibicuje Legii – jest zielona, a połowa Polonii – czerwona. Jak wiemy oba kluby nie pałają do siebie sympatią, a ich kibice dążą do tego, by mieszkać w sąsiedztwie swoich klubowych pobratymców. Dlatego też każdego roku, każdy mieszkaniec sprawdza ilu kibiców każdego klubu mieszka w odległości nie większej, niż 1km od niego (rysunek 1a). Jeśli przeważają kibice przeciwnego klubu, wtedy dany kibic przeprowadza się o 1km w dowolnym kierunku (rysunek 1b). Napisz program, który przeprowadzi symulację takiej sytuacji i pokaże ewolucję miasta w ciągu **x** lat.



Rysunek 1: Na zielono zaznaczeni są kibice Legii, na czerwono kibice Polonii. a) Zliczanie kibiców w obrębie 1km: zadowolenie zaznaczonego czarną kropką kibica wynosi tutaj -2. b) Ponieważ zadowolenie jest ujemne, kibic przeprowadza się o kilometr w dowolnym kierunku, czyli jego pozycja po przeprowadzce leży na przerywanym okręgu.

Organizacja kodu: Cały program powinien być umieszczony w katalogu Imie_Nazwisko. W katalogu musi być plik tekstowy o nazwie README, z zapisanym pełnym poleceniem do kompilacji i ewentualnymi wyjaśnieniami dotyczącymi uruchamiania i działania programu. Projekt musi być podzielony na klasy, a kod każdej klasy podzielony na deklarację w pliku nagłówkowym Klasa.h, oraz implementację w pliku o nazwie Klasa.cpp lub Klasa.cc lub Klasa.C. Dane składowe wszystkich klas powinny być prywatne.
Funkcja main() powinna być bardzo krótka.

Wymagania minimalne (na 50%): Napisanie programu opisującego powyższą sytuację, a w nim:

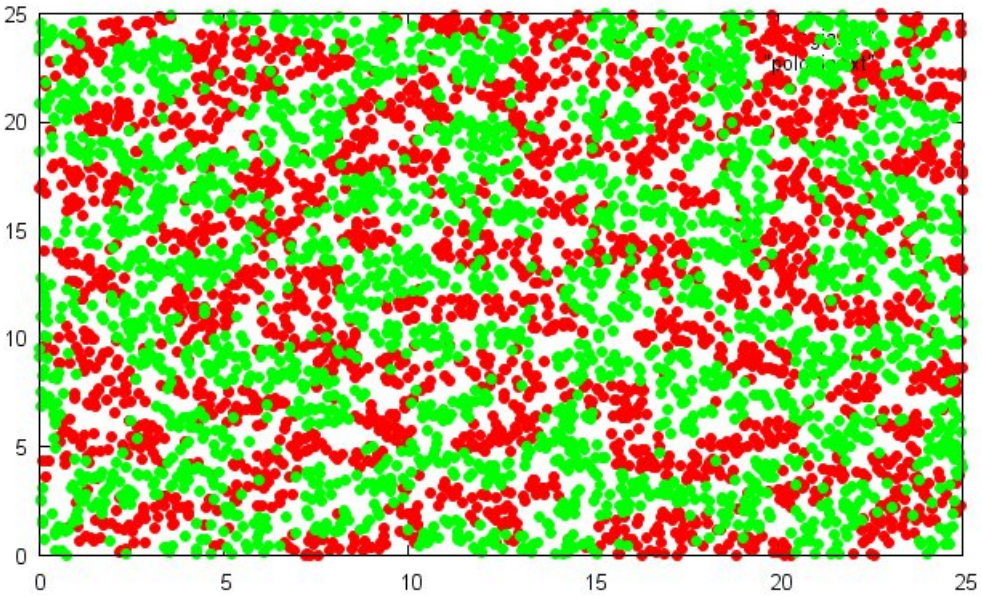
- stworzenie klasy `Miasto`, składającej się z parametrów opisujących samo miasto, vectora (lub tablicy) kibiców oraz z co najmniej następujących funkcji składowych (metod):
  - `zaludnij` – funkcji, która na podstawie argumentu `N` będącego całkowitą liczbą mieszkańców wylosuje miejsca zamieszkania `N` kibiców tak, że stosunek barw klubowych będzie wynosił 50:50. Przy każdym uruchomieniu programu symulacja powinna dawać inne wyniki.
  - `ewoluuj` – funkcji, która nie przyjmuje żadnych argumentów, a jej działanie polega na wykonaniu symulacji ewolucji miasta w trakcie jednego roku, czyli **sprawdzeniu dla każdego kibica jego osobistego zadowolenia oraz w przypadku ujemnego zadowolenia przeprowadzce danego mieszkańca**, przy czym będzie ona korzystała z metod klasy `Kibic`
  - `zapisz_i_narysuj` – funkcji, która zapisuje w dwóch kolumnach położenia `x` i `y` kibiców Legii i Polonii (kibiców Legii w pliku `legia.txt`, a kibiców Polonii w pliku `polonia.txt`), a następnie tworzy rysunek w programie `gnuplot`, pokazujący aktualne położenia kibiców obu drużyn.
    W tym celu funkcja powinna tworzyć plik tekstowy `skrypt.gp` z poleceniami dla programu `gnuplot`:

    ```
    plot "polonia.txt" with points pointtype 7,"legia.txt" with points pointtype 7
    pause -1 "Wciśnij ENTER"
    ```

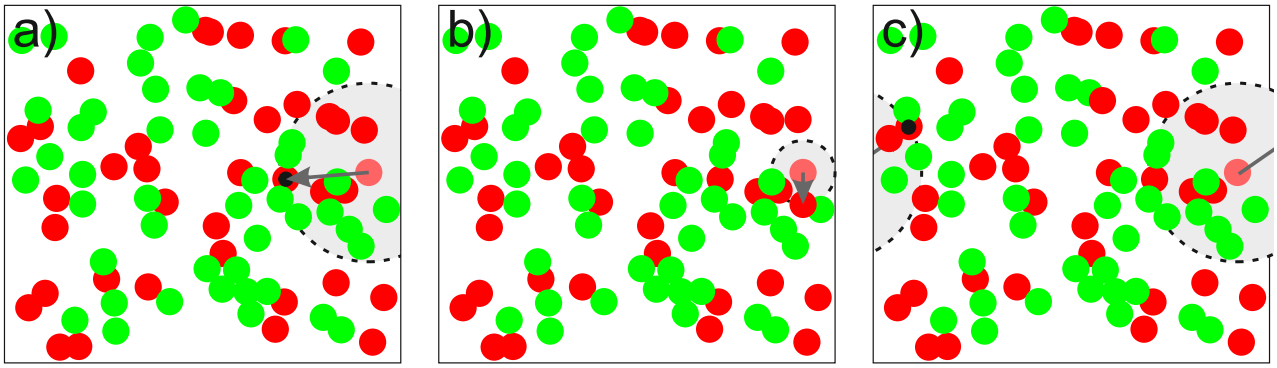
    (w ten sposób standardowo kibice Legii zostaną zaznaczeni na rysunku zielonymi kropkami, a kibice Polonii czerwonymi), a następnie wykonywać polecenie:
    `system("gnuplot skrypt.gp")`
- stworzenie klasy `Kibic`, składającej się z dwóch liczb zmiennoprzecinkowych opisujących położenie kibica(w zakresie 0-25) oraz parametru wybranego przez Ciebie typu, opisującego barwy klubowe. Ponadto klasa powinna zawierać co najmniej następujące funkcje składowe (metody):
  - `zadowolenie` – funkcji, która na podstawie argumentu `Miasto` liczy zadowolenie kibica z miejsca zamieszkania. Metoda powinna znaleźć liczbę kibiców jednego i drugiego klubu mieszkających w maksymalnej odległości 1km od badanego kibica i sprawdzić kibice którego klubu przeważają w danym sąsiedztwie. Powinna ona zwracać różnicę liczby kibiców klubu badanego kibica i liczbę kibiców przeciwnego klubu, patrz rysunek 1a).
  - `przeprowadzka` – funkcji poruszającej kibica w losowym kierunku o 1km.
- program powinien zapytać użytkownika o liczbę mieszkańców miasta **N** oraz liczbę **x** lat (kolejnych kroków symulacji), a następnie zaludnić miasto, wypisać poczatkowe średnie zadowolenie mieszkańca i pokazać położenia mieszkańców (poprzez uruchomienie funkcji zapisz_i_narysuj). Następnie powinien przeewoluować miasto, po każdym roku wyświetlając średnie zadowolenie mieszkańca i pokazując położenia mieszkańców. Program powinien sprawdzić poprawność wpisanych przez użytkownika danych (np. czy są dodatnie) i ew. wyświetlić odpowiedni komunikat. Przykładowy wygląd miasta po 5 latach przedstawiony jest na rysunku 2.

Wymagania na 75%: Jak wyżej, ale:

- należy zadbać, aby metoda `przeprowadzka` nie wyprowadzała mieszkańców z miasta, można wykorzystać jeden z trzech sposobów (rysunek 3): a) losować tak długo, dopóki docelowy punkt leży poza miastem, tak aby ostatecznie kibic przeprowadził się do miasta, b) zawężyć obszar przeprowadzki, tak aby kibic nie mógł przeprowadzić się za daleko (tzn. za miasto), c) użyć periodycznych warunków brzegowych, czyli *skleić* parami lewą i prawą, a także górną i dolną krawędź miasta, tak aby np. wyjście poza miasto z prawej strony oznaczało powrót do miasta z lewej strony (preferowane rozwiązanie o jak najwyższej literce).
- powinna istnieć możliwość zmiany promienia okręgu w którym sprawdzane jest zadowolenie mieszkańca i jednocześnie odległości o jaką mieszkaniec się przeprowadza
- klasa `Miasto` powinna posiadać konstruktor, który na podstawie liczby całkowitej `N` oznaczającej całkowitą liczbę mieszkańców oraz liczby zmiennoprzecinkowej `p` oznaczającej procent kibiców zielonych (np. 0.5 to stosunek 50:50) stworzy zaludnione już miasto



Rysunek 2: Przykładowy wygląd miasta po 5 latach



Rysunek 3: Sposoby manipulowania kibicem, aby przeprowadził się tylko w ramach miasta. a) Losowanie nowego miejsca zamieszkania tak długo, aż nowe miejsce zamieszkania zostanie ustalone jako wewnątrz miasta. b) Losowanie nowego miejsca zamieszkania w zawężonym zakresie, tak aby możliwy obszar przeprowadzki nie wychodził poza miasto. c) Zastosowanie periodycznych warunków brzegowych, czyli *sklejenie* lewej i prawej oraz górnej i dolnej krawędzi miasta

- klasa `Kibic` powinna posiadać konstruktor, który automatycznie wylosuje miejsce zamieszkania kibica, a jego klub przypisze na podstawie argumentu konstruktora wybranego przez Ciebie typu
- klasa `Miasto` powinna posiadać metodę, która na podstawie obiektu typu `string` wpisywanego przez użytkownika z klawiatury w funkcji `main`, powinna zaludnić miasto poprzez wczytanie pliku, którego nazwa zawiera się w podanym `string`u. Plik zawiera trzy kolumny: pozycję x, pozycję y, klub kibica w postaci obiektu typu `string`: *Legia* lub *Polonia*. Każdy wiersz to osobny kibic np:

1.0	15.2	Polonia
5.4	17.5	Legia
13.1	7.2	Legia
24.5	12.2	Polonia

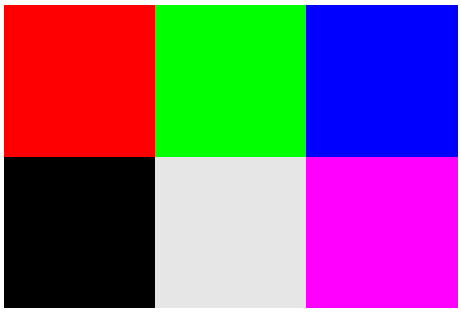
  Zakładamy, że plik jest napisany poprawnie.
- Program po uruchomieniu powinien zapytać użytkownika czy chciałby losować mieszkańców czy wczytać plik – jeśli użytkownik odpowie że losować, to program ma wczytać z klawiatury liczbę `N` mieszkańców miasta, stosunek liczby kibiców z obu klubów `p`, liczbę `x` lat (kolejnych kroków symulacji) oraz promień okręgu `r` w którym sprawdza się zadowolenie mieszkańca (i jednocześnie odległość o jaką

mieszkaniec się przeprowadza). Następnie powinien zaludnić miasto oraz je przeewoluować `x` razy, po każdym roku wyświetlając średnie zadowolenie mieszkańca. Natomiast jeśli użytkownik zdecydował sie wczytać plik to program powinien wczytać z klawiatury liczbę `x` oraz nazwę pliku z kibicami, a następnie zaludnić miasto zgodnie z plikiem i przeewoluować je, po każdym roku wyświetlając średnie zadowolenie mieszkańca.

Wymagania na 100%: Jak wyżej, ale:

- zaimplementuj obsługę źle napisanego pliku z kibicami do wczytania, np. zła nazwa klubu (słowo inne niż *Polonia* lub *Legia*), położenie kibica poza miastem czy niepełne dane. W przypadku złej nazwy klubu zapytaj użytkownika czy przypisać kibica do konkretnego klubu, czy pominąć go, czy zakończyć program;
- powinna być możliwość uruchomienia programu z wszystkimi parametrami przekazywanymi do programu jako argumenty funkcji `main`;
- klasa `Miasto` powinna dodatkowo zawierać metodę `narysuj` o argumentach: `string` będący nazwą pliku, liczba całkowita `M` oraz liczba zmiennoprzecinkowa `R`. Metoda powinna stworzyć nowy plik o podanej nazwie, z rozszerzeniem `.ppm`, a w nim zapisany obrazek o rozdzielczości `M`×`M`, na którym będą zaznaczeni kibice w postaci kropek o promieniu `R` pikseli i odpowiednim kolorze: zielonym (kibice Legii) lub czerwonym (kibice Polonii).
  Wyjaśnienie: obrazki w formacie PPM [1] są to pliki tekstowe zawierające trzy linijki nagłówka oraz trójki liczb opisujące kolorowe piksele. Przykładowy plik tekstowy i odpowiadający mu obrazek wyglądają jak poniżej:

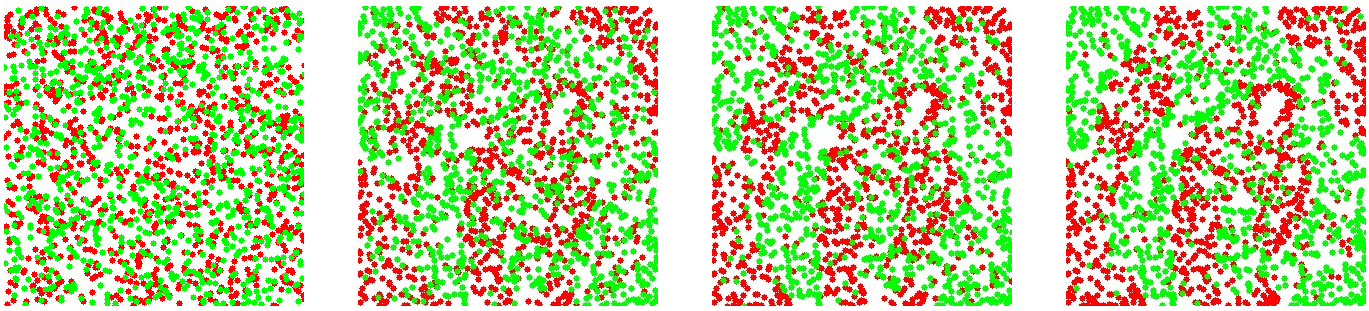
```
P3
3 2
255
255 0 0
0 255 0
0 0 255
0 0 0
230 230 230
255 0 255
```



W pliku znajdują się kolejno:

1. nagłówek: P3 – oznacza kolorowy obrazek zapisany tekstowo (dla nas niezmienne)
2. dwie liczby całkowite oznaczające szerokość i wysokość obrazka mierzoną w pikselach, tutaj obrazek o szerokości 3 i wysokości 2 pikseli
3. maksymalna wartość koloru – standardowo 255, tego nie zmieniamy – oznacza ona rozdzielczość w kolorze, możliwe wartości zawierają się w tym przykładzie pomiedzy 0 a 255
4. szerokość×wysokość×3 liczb całkowitych odzielonych białymi znakami – każda kolejna trójka liczb oznacza piksel w formacie RGB, np. piksel czerwony to: 255 0 0, a zielony: 0 255 0

- Program powiniem zapisywać obrazek mapy miasta co 5 lat ewolucji (rysunek 4).



Rysunek 4: Przykładowa ewolucja miasta

Wymagania na punkty powyżej 100%: dodatkowe punkty można uzyskać poprzez napisanie bezbłędnego programu spełniającego wszystkie powyższe kryteria i rozszerzającego powyższą specyfikację, poprzez np.:

- zastosowanie dziedziczenia klas, polimorfizmu i metod wirtualnych, tak aby mieć dwa różne rodzaje kibiców zapisanych w postaci dwóch różnych klas dziedziczących z klasy `Kibic`, np. jedna podklasa liczy zadowolenie z koła o promieniu 1km jak wyżej, zaś druga podklasa liczy zadowolenie na podstawie kwadratu o boku 1km.
- inne rozszerzenia programu skonsultowane wcześniej z wykładowcą.