Programowanie obiektowe dla AiR - Kolokwium 1 - 16 kwietnia 2024 r.

Rozwiązanie kolokwium, w postaci plików źródłowych `*.cpp` i nagłówkowych `*.h`, umieść w Moodle
`https://moodle.umk.pl/mod/assign/view.php?id=207481`
Zaliczenie wymaga wykonanie co najmniej 50% programu zgodnie z poniższym opisem.

Zadanie: Błądzenie losowe.
Napisz program, który przeprowadzi symulację błądzenia pijaków po mieście i wyznaczy średnią drogę przebytą przez pijaka. Program implementuje dwie klasy `Pijak` i `Symulacja` zgodnie z poniższą specyfikacją:

Klasa `Pijak` reprezentuje osobę błądzącą losowo po ulicach miasta. Zakładamy, że miasto składa się z ulic tworzących nieskończoną siatkę na płaszczyźnie. Odległość pomiędzy kolejnymi skrzyżowaniami wynosi 1. Po ulicach porusza się pijak w ten sposób, że na każdym skrzyżowaniu wybiera losowo jedną z czterech dróg (włączając tę, którą przyszedł) i porusza się dalej prosto aż do kolejnego skrzyżowania (zob. rysunek na odwrocie).
Pijak posiada następujące atrybuty:
- `x`, `y`: współrzędne położenia na płaszczyźnie

Pijak udostępnia następujące operacje:
- konstruktor domniemany tworzący pijaka w pozycji początkowej $(0, 0)$
- konstruktor tworzący pijaka w pozycji o współrzędnych (x, y) podanych w argumentach
- funkcję składową `Ruch()`, która powoduje ruch pijaka w kierunku kolejnego skrzyżowania. Każda z 4 dróg może być wybrana z takim samym prawdopodobieństwem $\frac{1}{4}$. Wywołanie funkcji powoduje zmianę współrzędnych położenia pijaka
- funkcję składową `Odleglosc()`, która zwraca liczbę rzeczywistą równą odległości pijaka od środka układu współrzędnych $(0, 0)$ wyznaczoną ze wzoru
$$\sqrt{x^2 + y^2}$$
- przeciążenie operatora `<<`, który umieszcza w strumieniu wyjściowym `std::ostream` informację o aktualnej pozycji pijaka, np.: `"(1 ,5)"`

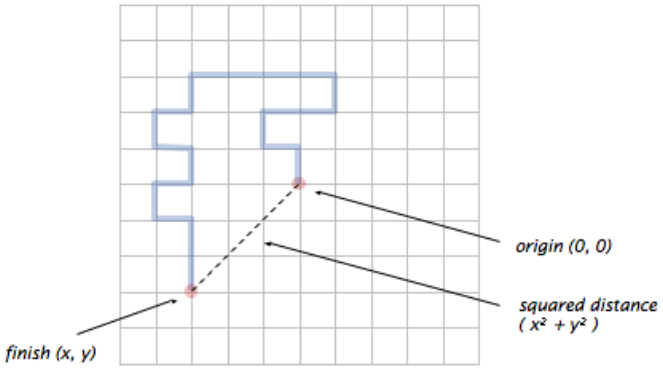
Klasa o nazwie `Symulacja` umożliwia przeprowadzenie doświadczenia, w którym po mieście błądzi N pijaków jednocześnie. Symulacja posiada następujące atrybuty:
- `ile`: ilość pijaków, liczba całkowita
- `lista`: adres do tablicy zawierającej tablicę pijaków (obiektów typu `Pijak`)
- `tura`: aktualny krok symulacji, liczba całkowita

Symulacja udostępnia następujące operacje:
- konstruktor z jednym argumentem typu całkowitego określającym ilość pijaków biorących udział w symulacji. Tworzona jest tablica pijaków w pozycji domyslnej $(0, 0)$. Numer tury ustalany jest na 0.
- konstruktor kopiujący
- funkcję składową `Ruch()`, która wykonuje pojedynczą turę symulacji, w której każdy z pijaków wykonuje ruch w losowym kierunku. Wartość tury wrasta o 1
- funkcję składową `Srednia()`, która zwraca średnią kwadratową (RMS) odległości pijaków od środka układu współrzędnych daną wzorem
$$\sqrt{\frac{1}{n}\sum_{i=1}^{n} d_i^2}$$ gdzie d_i jest odległoscia i-tego pijaka
- przeciążenie operatora `<<` umieszczającego w strumieniu wyjściowym `std::ostream` listę pijaków, ich pozycje i przebyty dystans w postaci takiego komunikatu

```
Ilosc pijakow: 3
Numer tury: 10
Polozenie i odleglosc
1: (3, -1), 3.16228
2: (4, 2), 4.47214
3: (2, 2), 2.82843
```

Wykorzystaj klasy `Pijak` oraz `Symulacja` do stworzenia programu realizującego następujące operacje:
1. użytkownik podaje w konsoli ilość pijaków N i ilość kroków K symulacji
2. program tworzy symulację zawierająca N pijaków ustawionych w pozycji poczatkowej $(0, 0)$
3. wykonywane jest K krokow symulacji
4. wypisywana jest pozycja wszystkich pijaków oraz wartość średniej kwadratów (RMS) odległości na którą oddalili sie pijacy
5. program wypisuje też wartość oczekiwaną RMS, która wynosi $\sqrt{K}$

Pamiętaj o hermetyzacji klas (pola składowe powinny być prywatne), jeśli uznasz za potrzebne to zaimplementuj dodatkowe mechanizmy, np. destruktory, przeciążane operatory, deklaracje przyjaźni, itp.



Rysunek 1: Błądzenie losowe pijaka po mieście

Przykładowe działanie programu:

```
Ilu pijakow? 10
Ile kroków symulacji? 100

Ilosc pijakow: 10
Numer tury: 100
Polozenie i odleglosc
1: (0, -8), 8
2: (2, 4), 4.47214
3: (4, -4), 5.65685
4: (14, 4), 14.5602
5: (-7, 7), 9.89949
6: (3, -3), 4.24264
7: (-2, 8), 8.24621
8: (2, 14), 14.1421
9: (-9, 13), 15.8114
10: (-8, 4), 8.94427

Srednia kwadratowa odleglosc w symulacji: 10.2078
Srednia kwadratowa odleglosc (wart. oczekiwana): 10
```