Simulácia robotického vysávača

23. 5. 2024

Vašou úlohou je vytvoriť simuláciu náhodného pohybu robotického vysávača v jednej miestnosti, pričom skúmame, ako dlho trvá vysávaču pozbierať prach. Každá simulácia obsahuje jednu miestnosť a jeden vysávač. V skripte pre implementáciu simulácie (`0523-1.py`) nájdete dve triedy:

Room

Definuje miestnosť: určí jej rozmery a reprezentuje ju ako dvojrozmerné pole:

- `width` (int) – šírka miestnosti
- `length` (int) – dĺžka miestnosti
- `room` (list) – dvojrozmerné pole reprezentujúce miestnosť s rozmermi `width` × `length`; na začiatku simulácie sa na každej pozícii nachádza `0`

Doplňte metódy do triedy podľa nasledovných špecifikácií:

- **`add_dust(count)`** – pridá do miestnosti `count` častíc prachu na náhodné pozície, pričom `count` je kladné celé číslo; viaceré častice môžu byť pridané na tú istú pozíciu, pridanie reprezentujte inkrementáciou hodnoty na danej pozícii v dvojrozmernom poli `room` (čísla v poli teda vyjadrujú počet častíc prachu na danej pozícii)

- **`has_position(pos)`** – funkcia určí, či pozícia s danými súradnicami existuje v miestnosti; vstupný parameter `pos` je dvojica súradníc `x` a `y`, ak niektorá z nich je záporná alebo väčšia ako príslušný rozmer miestnosti, funkcia vráti `False`, v opačnom prípade `True`

- **`has_dust(pos)`** – funkcia vráti booleovskú hodnotu, ktorá určí, či sa na danej pozícii `pos` nachádza prach (hodnota je väčšia ako `0`); `pos` je dvojica hodnôt `x` a `y`, ak daná pozícia neexistuje, funkcia vygeneruje `ValueError`

- **`pickup_dust(pos)`** – funkcia reprezentuje odstránenie jednej častice prachu z danej pozície, ak pozícia existuje a obsahuje prach (hodnota uložená na pozícii je väčšia ako `0`); vstupný parameter `pos` je dvojica súradníc `x` a `y`; odstránenie prachu je reprezentované dekrementáciou hodnoty o `1`

- **`is_clean()`** – funkcia určí, či je miestnosť čistá, teda na žiadnej pozícii sa nenachádza hodnota väčšia ako `0`; funkcia nemá vstupný parameter

VacuumCleaner

Trieda reprezentuje vysávač, ktorý je definovaný tromi hodnotami:

- `current_position` (tuple) – pozícia vysávača v miestnosti; je to dvojica celočíselných hodnôt, ktoré reprezentujú súradnice `x` a `y`
- `possible_directions` (list) – zoznam všetkých možných smerov pohybu vysávača
- `room` (objekt typu `Room`) – odkaz na objekt reprezentujúci miestnosť

V triede potrebujete definovať iba jednu funkciu, **`move(direction)`**:

1. ak sa na aktuálnej pozícii vysávača nachádza častica prachu, zavolajte príslušnú metódu z triedy `Room` aby ste ju odstránili; funkcia okamžite po tom ukončí vykonávanie
2. na základe smeru `direction` vypočítajte novú pozíciu vysávača z aktuálnej pozície (`current_position`); ak `direction` má hodnotu, ktorá nepatrí do podporovaných smerov, vygeneruje sa `ValueError`
3. ak nová pozícia existuje (nachádza sa v miestnosti), aktualizujte hodnotu aktuálnej pozície (`current_position`) – posuňte teda vysávač na novú pozíciu

Skript obsahuje metódu `simulate_cleaning()`, ktorá vykoná niekoľko simulácií vysávača. Metóda má nasledovné parametre:

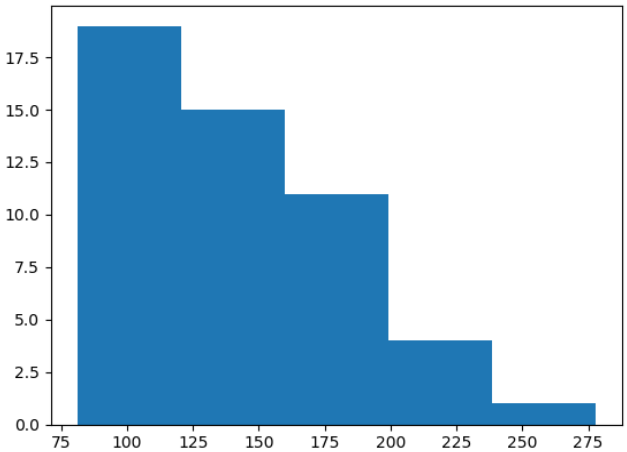
- `room_dimensions` (tuple) – rozmery miestnosti (šírka a dĺžka)
- `dust_count` (int) – počet častíc prachu, ktoré treba pridať do miestnosti pred spustením simulácie
- `simulation_no` (int) – počet simulácií, ktoré metóda má vykonať

Implementujte metódu **`simulate_cleaning()`** podľa nasledovnej špecifikácie:

- metóda má jednu návratovú hodnotu – zoznam, ktorý obsahuje počet potrebných krokov na úplné očistenie miestnosti v jednotlivých simuláciách
- vykonajte `simulation_no` simulácií, pričom každá simulácia má nasledovnú štruktúru:
    1. vytvorte novú miestnosť s požadovanými rozmermi a pridajte potrebné častice prachu
    2. vytvorte vysávač na náhodnej pozícii v miestnosti
    3. kým miestnosť nie je očistená, vyberte náhodný smer vysávača a zavolajte príslušnú metódu pre pohyb vysávača
    4. zaznamenajte, koľko krokov bolo potrebných na očistenie miestnosti a pridajte hodnotu do zoznamu, ktorý na konci vykonávania vracia metóda

Vo funkcii main() odpovedzte na otázku:
Aká je distribúcia potrebných krokov na očistenie miestnosti?

Pre získanie odpovede vykonajte Vami určený počet simulácií. Výsledky znázornite pomocou histogramu; graf musí mať názov, a pomenované osi. Samotný graf môže vyzerať nasledovne:



Pri riešení dodržujte nasledujúce zásady:

- riešenie môžete rozšíriť ľubovoľnými metódami
- pri riešení nemusíte postupovať presne podľa návodu, mali by ste ale ponechať funkcionalitu
- predpripravený skript môžete ľubovoľne opravovať, nemali by ste ale meniť funkcionalitu
- nepristupujte priamo k členským premenným objektov a tried
- hlavná funkcia môže obsahovať iba volanie funkcie `main()`

Dĺžka kódu (bez komentárov): cca. 110 riadkov.