Informatyka I – projekt zaliczeniowy S1 2020/2021

1130-00000-ISP-1006

Projekt numer 15. „Totolotek” – symulator maszyny losującej Lotto

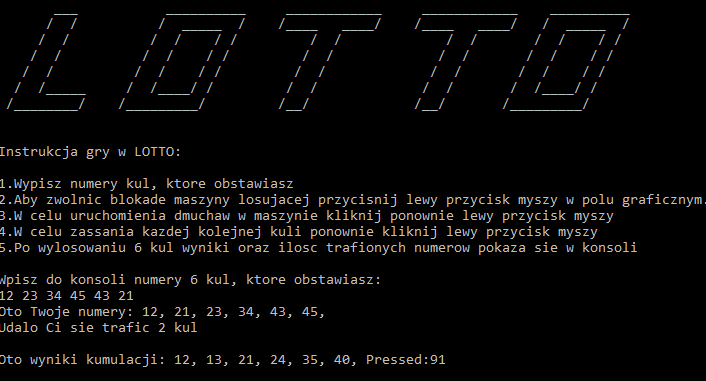
*Spis treści:*

1. Ogólne założenia projektu
2. Przebieg działania programu
3. Struktura programu, funkcje
4. Założenia modelu maszyny losującej zawarte w programie
5. Rozwiązanie najważniejszych problemów niezbędnych dla prawidłowego działania programu
6. Wady i możliwe dalsze usprawnienia programu
7. **Ogólne założenia programu**

Program jest uproszczoną symulacją maszyny losującej kule w lotto. W pojemniku znajduje się 49 kul, zadaniem programu jest wylosowanie 6 z nich. Początkowe rozmieszczenie kul w naszym programie jest losowe, aby zwiększyć losowość wyników – w realnej maszynie na wyniki wpływ ma bardzo duża ilość czynników.

1. **Przebieg działania programu**

Po wyświetleniu logo, użytkownik ma do przeczytania instrukcję do korzystania z programu, następnie program oczekuje od użytkownika wpisania 6 liczb, które użytkownik „obstawia” na swoim kuponie lotto. Po otwarciu okna graficznego, aby wylosować rozmieszczenie kul w boksie należy kliknąć lewy przycisk myszy. Kolejnymi krokami są włączenie dmuchaw oraz otwarcie dyszy zasysającej kule, użytkownik musi to zrobić klikając lewy przycisk myszy. Po każdym jednorazowym wylosowaniu kuli użytkownik musi ponownie otworzyć dyszę zasysającą. Po wylosowaniu 6 kul program w oknie graficznym pokaże wylosowane numery wraz z kolorem kul. W konsoli użytkownik może zobaczyć, ile numerów pokrywa się z jego losem.



1. **Struktura programu, funkcje**

void initPosition(double T[49][5], int N); - losowe rozmieszczenie kul na planszy

void collideWall(double t[49][5], int N); - algorytm symulujący zderzenia ze ścianami

void collideBall(double t[49][5], int N); - algorytm symulujący zderzenia z kulami

void move(double t[49][5], int N); - zmiana współrzędnych piłki po uwzględnieniu wszystkich sił i zależności

void display(double t[49][5], int N); - wyświetlanie okna graficznego i maszyny

void dmuchawalewa(double t[49][5], int n); - algorytm zmieniający prędkość piłek na skutek działania dmuchawy

void dmuchawaprawa(double t[49][5], int n); - analogicznie do dmuchawy lewej

void sortuj(double t[49][5], int n); - sortowanie, pomagające usunąć kule które zostały wylosowane z zasobnika

void sortuj1(int t[6], int n); - sortowanie wyników

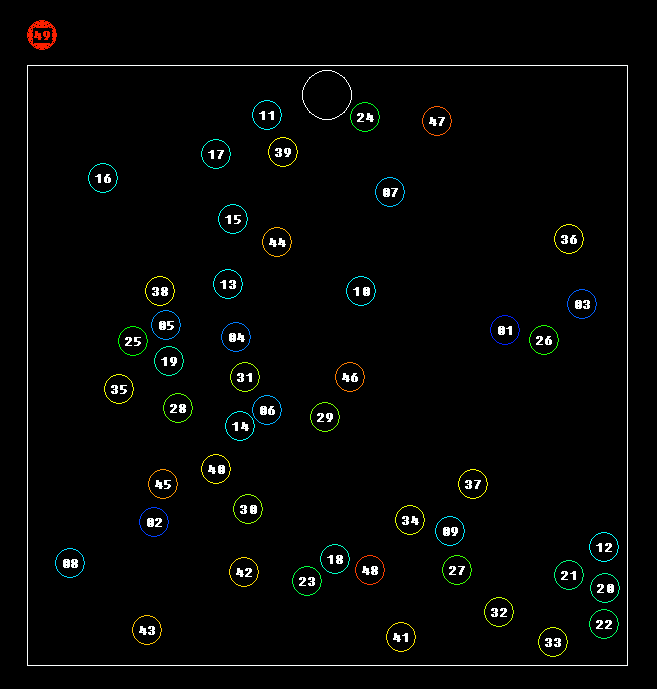
void zasysanie(double t[49][5], int n, int w[6]); - algorytm zmieniający prędkości piłek na skutek różnicy ciśnień po otwarciu dyszy zasysającej

void kula(double x, double y, int r); - wypełnianie kolorem kul wylosowanych

void instrukcja(); - wyświetlanie logo i instrukcji

1. **Model maszyny losującej oraz założenia zawarte w programie**

Aby ruch kul oraz wyniki były różne przy każdym losowaniu rozmieszczenie kul w programie jest inicjowane poprzez pseudo losowanie ich współrzędnych. Maszyna w programie składa się z dwóch dmuchaw umieszczonych w dwóch dolnych rogach pojemnika, ich siła działania na kule jest zależna od odległości pomiędzy nimi, a wylotem dmuchawy. W każdym momencie działania programu na każdą kulę działa siła spowodowana przez dwie dmuchawy oraz grawitację. Po uruchomieniu zasysania wpływ na prędkość i kierunek lotu piłek ma również siła imitująca zasysanie, kieruje wszystkie kule w miejsce zasysania – im bliżej wylotu, tym przyspieszenie piłek jest większe.



Podsumowując, ruch kul zmienia się na skutek:

- kolizji z innymi kulami

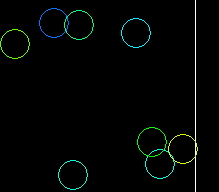
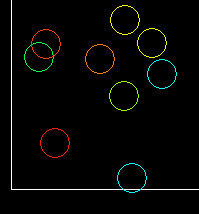
- kolizji ze ścianami

- działania siły grawitacji

- działania dwóch dmuchaw

- dyszy zasysającej

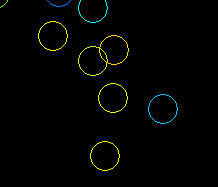
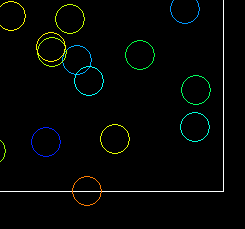
1. **Rozwiązanie największych problemów niezbędnych do prawidłowej symulacji**
2. Algorytm zderzeń z innymi piłkami, zaczerpnięty z Laboratorium 5 i przystosowany do naszej symulacji nie uwzględniał sytuacji zderzenia ze sobą w jednym momencie więcej niż dwóch piłek, co przy liczbie 49 piłek i niewielkim oknie graficznym jest nieuniknione:



Rozwiązanie:

Gdy piłki po zmianie współrzędnych spowodowanej prędkością, znajdą się w odległości od siebie mniejszej niż 2 promienie – najpierw są rozsuwane, wracają na współrzędne, tak aby się ze sobą stykały, a nie nachodziły na siebie i dopiero wtedy zwrot ich prędkości jest zmieniany.

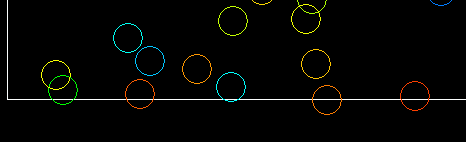
1. Ten sam algorytm przy dużych prędkościach piłek i dużej ilości zderzeń powodował zlepianie się kul – zwrot prędkości zmieniał się kilkukrotnie, ponieważ w ciągu jednego przebiegu pętli kule nie zdążyły odsunąć się na tyle, aby znaleźć się poza obszarem drugiej kuli:



Rozwiązanie:

Tak jak problem pierwszy.

1. Algorytm zderzeń z ścianami nie uwzględniał sytuacji zderzenia z ścianą oraz inną kulą w jednym momencie:



Rozwiązanie:

Analogicznie do problemów powyżej – kula najpierw jest przesuwana tak, aby cała znajdowała się w zasobniku maszyny, dopiero potem działa algorytm odbicia.

1. Algorytm odbicia od ściany nie uwzględniał sytuacji, gdy kula znajdzie się w rogu planszy.

Rozwiązanie:

Dodatkowe warunki sprawdzające czy piłka nie znajduje się w rogu planszy.

1. **Możliwe dalsze usprawnienia programu:**
2. Optymalizacja kodu – m. in. lepszy podział na funkcje oraz likwidacja zdublowanych instrukcji
3. Uroszczenie maina, i podzielenie go na kolejne funkcje tak aby funkcja główna wyglądała przejrzyście.