Курсова работа по ПИК2

Даниил Доцев, 012219004, АИУТ

Задание:

6.07.4. Да се реализира движение на материална точка в зададено правоъгълно поле. Да може да се задава посока на движение, скорост, маса и коефициент на триене. Ударите в стените са идеално еластични. Програмата работи в екранен текстов режим. Да се реализира цветно.

Програмен код: https://github.com/dddotsev/MovingPoint/

Език: C

Използвани библиотеки:

- windows.h – за „рисуване по екрана“
- stdio.h – за четене и писане в конзолата
- math.h – математически функции
- stdbool.h – bool тип данни

Дефинирани стойности:

- π
- g – ускорение към полето, симулира земно притегляне. Използвана е случайна стойност

Създадени структури:

- FLOATING_COORD – X, Y координати в тип с плаваща запетая

Константи:

- FLOATING_COORD pos_start – начална позиция на топчето
- double timestep – време в секунди, което е минималната единица, за която се превят изчисленията
- COORD hide_coord – позиция, на която да се „скрие“ курсора, докато не изписва нищо

Променливи:

- double mass – маса на топчето
- double u – коефициент на триене
- double speed – скорост на топчето
- FLOATING_COORD pos – текуща позиция на топчето
- double angle – текущ ъгъл на топчето спрямо абсцисата
- font_ratio – съотношение на текущия избран шрифт, X/Y
- short actual_width, actual_height – широчина и височина на прозореца

Функции:

COORD toCoord(FLOATING_COORD fc)

Преобразува от целочислени координати в координати с плаваща запетая

bool move(HANDLE hOutput)

Преместване на точката с една стъпка. Връща false, ако топчето е спряло и true – ако не е

```
double acc = - u * mass * g;
double dist = speed * timestep + acc * timestep * timestep / 2;
speed += acc * timestep;
```

Изчисляава се намалението на ускорението, изминатото разстояние за една стъпка и новата скорост.

```
if (speed < 0) {
    speed = 0;
    return false;
}
```

Ако скоростта стане под нула, означава че топчето трябва да е спряло.

```
double dx, dy;
if (angle ≥ 0 && angle < PI / 2) {
    dx = cos(angle) * dist;
    dy = -sin(angle) * dist;
}
else if (angle ≥ PI / 2 && angle < PI) {
    double current_angle = angle - PI / 2;
    dx = -sin(current_angle) * dist;
    dy = -cos(current_angle) * dist;
}
else if (angle ≥ PI && angle < PI + PI / 2) {
    double current_angle = 3 * PI / 2  - angle;
    dx = -cos(current_angle) * dist;
    dy = sin(current_angle) * dist;
}
else {
    double current_angle = PI * 2 - angle;
    dx = cos(current_angle) * dist;
    dy = sin(current_angle) * dist;
}
FLOATING_COORD new_pos = {pos.X + dx, pos.Y + dy};
```

Изчисляват се промените в координатите в изминалата стъпка (dx, dy) и се изчисляват новите координати.

```
COORD old_coord = toCoord(pos);
COORD new_coord = toCoord(new_pos);
pos = new_pos;

if (old_coord.X == new_coord.X && old_coord.Y == new_coord.Y) {
    return true;
}
```

Координатите се преобразуват в целочислени. Ако старите и новите координати съвпадат, няма нужда да се правят допълнителни изчисления, понеже на екрана се визуализират целочислени координати и нищо не се е променило.

```
if (new_coord.X < 0) {
    new_coord.X = 0;
    angle = 3 * PI - angle;
}
else if (new_coord.Y < 0) {
    new_coord.Y = 0;
    angle = 2 * PI - angle;
}
else if (new_coord.X ≥ actual_width + 1) {
    new_coord.X = actual_width;
    angle = PI - angle;
}
else if (new_coord.Y ≥ actual_height + 1) {
    new_coord.Y = actual_height;
    angle = 2 * PI - angle;
}
```

Проверка за удар със стена.

```
SetConsoleCursorPosition(hOutput, old_coord);
WriteConsole( hOutput, " ", 1, NULL, NULL );
```

Изчистване на старата позиция на топчето.

```
SetConsoleCursorPosition(hOutput, new_coord);
WriteConsole( hOutput, "o", 1, NULL, NULL );
```

Изписване на новата позиция на топчето.

```
SetConsoleCursorPosition(hOutput, hide_coord);
```

Преместване на курсора на позиция, която не бие на очи.

void main(void)

Входна точка на програмата.

```
printf("Enter start angle in radians relative to horizonal axis:\n");
scanf("%lf", &angle);
printf("Enter start speed:\n");
scanf("%lf", &speed);
printf("Enter mass:\n");
scanf("%lf", &mass);
printf("Enter friction coefficient:\n");
scanf("%lf", &u);

system("cls");
```

Въвеждане на параметрите и изчистване на екрана.

```
pos = pos_start;

HANDLE hOutput = (HANDLE)GetStdHandle(STD_OUTPUT_HANDLE);

CONSOLE_SCREEN_BUFFER_INFO SBInfo;

GetConsoleScreenBufferInfo(hOutput, &SBInfo);

actual_width = SBInfo.srWindow.Right;
actual_height = SBInfo.srWindow.Bottom;

SetConsoleTextAttribute(hOutput,
    FOREGROUND_INTENSITY | FOREGROUND_GREEN);

CONSOLE_FONT_INFO info;
GetCurrentConsoleFont(hOutput, false, &info);
COORD font_size = GetConsoleFontSize(hOutput, info.nFont);

font_ratio = (float)font_size.X / font_size.Y;
```

Инициализиране на програмата и неободими стойности. Задаване на цвят на топчето.

```
while (move(hOutput)) {
    Sleep(timestep * 1000);
}
```

Докато move функцията връща true спи timestep секунди.

```
scanf("\n");
```

След спиране на топчето програмата изчаква въвеждане преди да спре.

Входни данни:

- Текущ ъгъл на топчето спрямо абсцисата. Примерна стойност: 0.8
- Скорост на топчето. Примерна стойност: 40
- Маса на топчето. Примерна стойност: 20
- Коефициент на триене. Примерна стойност: 0.1

Допълнение:

За оптимална визуализация трябва да се избере квадратен шрифт на конзолата.