INFORMATYKA I: INSTRUKCJA 9

1 Oblężenie

Do poprzednich zajęć poznaliśmy już wszystkie instrukcje zaplanowane na ten kurs języka C. Czas przejść do kilku nieco bardziej dojrzałych przykładów. Dziś zajmiemy się oblężeniem zamku. Twoim zadaniem jest napisać program, który będzie symulował ostrzał zamku przez kule armatnie (kule lecą wg wzorów na rzut ukośny). Program powinien kolejno:

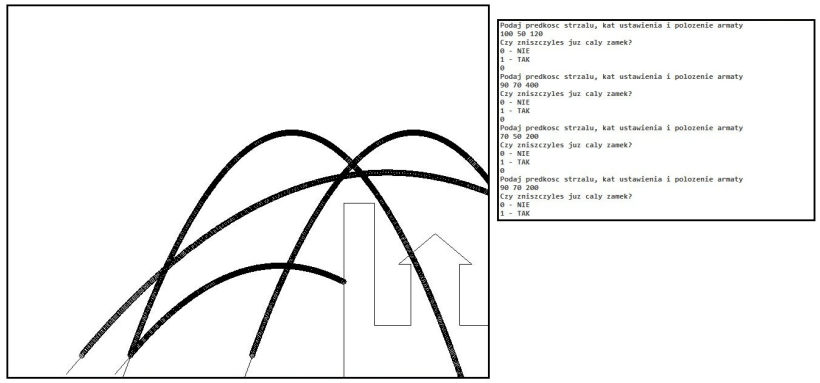
- Przygotować okno graficzne o wymiarach 800 x 600.
- Narysować zamek, który będziemy burzyć (kod funkcji rysującej zamek możesz znaleźć na końcu instrukcji).
- W pętli:
  - wczytywać z klawiatury wartość prędkości wylotowej kuli V_0, kąta odchylenia lufy od poziomu α oraz położenie armaty wzdłuż współrzędnej x (wylot armaty zawsze ma się znajdować 50 jednostek powyżej poziomu ziemi).
  - ustawiać i rysować schematycznie armatę jako jedną linię (należy przy tym sprawdzać, czy w danym miejscu da się postawić armatę - mur zamku znajduje się w obszarze $x \geq 550$; dla uproszczenia radzimy przy tym rysować armatę, jako linię o górnym wierzchołku w punkcie *(x, 50)* i dolnym wierzchołku zależnym od kąta wychylenia).
  - wykonywać strzał kulą i rysować jej tor za pomocą gęsto rozłożonych okręgów o małym promieniu - użyj funkcji `animate` w celu uzyskania efektu płynnego ruchu kuli (przykład użycia funkcji `animate` na końcu instrukcji). Tor rzutu ukośnego dany jest wzorami:

$$x(t) = x_0 + V_0 sin(\alpha)t \tag{1}$$

$$y(t) = y_0 + V_0 cos(\alpha)t - \frac{gt^2}{2}. \tag{2}$$

Program powinien cały czas w pętli sprawdzać, czy kula nie uderzyła w pierwszą, pionową ścianę zamku. Jeśli tak, kula ma się na niej zatrzymać, a pętla ulec przerwaniu. W przeciwnym razie program ma tak długo rysować ruch kuli, aż ta znajdzie się na ujemnych wartościach współrzędnej y. Ściana zamku zawiera się w przedziale $550 \leq x \leq 600$ oraz jej górna powierzchnia ma współrzędną $y = 300$. Pamiętaj również o przeliczeniu współrzędnych fizycznych (pionowej) na współrzędną w układzie współrzędnych ekranu oraz przeliczeniu kąta podawanego w stopniach na kąt podany w radianach na potrzeby funkcji trygonometrycznych.

  - Po wykonanym strzale program zapyta, czy udało się trafić w zamek. Jeśli odpowiesz twierdząco, zakończy działanie programu. Jeśli przecząco, pozwoli wybrać nowe parametry strzału i będzie powtarzał tę operację aż do chwili uzyskania satysfakcjonującego cię wyniku.



Rysunek 1: Oczekiwany efekt działania programu

Wskazówka

W niniejszym programie w ogóle nie używaj funkcji `wait()`. Program z wykonaniem i tak będzie czekać na podanie nowego zestawu danych do funkcji `scanf()`.

Oczekiwany wynik

Spodziewamy się, że wynikiem działania twojego programu będzie efekt zbliżony do tego pokazanego na Rysunku 1.

Dodatki

Funkcja `animate` tak naprawdę jedynie spowalnia wykonywanie pętli `while` tak, aby kółka nie rysowały się zbyt szybko. Przykład jej użycia wygląda następująco:

```
while(animate(100)) // Wartosc 0 oznacza najszybsze mozliwe
                    // wykonanie (brak opoznienia)
{
   // Oblicz tu nowe wspolrzedne x i y
   // Narysuj okrag w odpowiednim miejscu
   // Zwieksz parametr t
   // Zawrzyj warunek przerwania petli, gdy kula uderza
   // w sciane lub spada na y < 0
}

void RysujZamek()
{
   line(550, 300, 550, 600);
   line(550, 300, 600, 300);
   line(600, 300, 600, 500);
   line(600, 500, 660, 500);
   line(660, 500, 660, 400);
   line(660, 400, 640, 400);
   line(640, 400, 700, 350);
   line(700, 350, 760, 400);
   line(760, 400, 740, 400);
   line(740, 400, 740, 500);
   line(740, 500, 800, 500);
}
```