Instrukcja projektowa; projekt numer 1

Podstawy Programowania 2015/2016, kierunek Informatyka

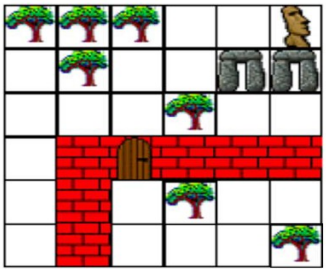
autor: Krzysztof Bruniecki[1]

wersja z dnia: 2015-11-12

Projekt: ASCII Labirynt

Cel

Zadanie polega na implementacji programu konsolowego wyświetlającego labirynt w postaci rzutu z góry oraz w postaci widoku "pseudo 3D" FFP (ang. first person perspective), z odwzorowaniem perspektywy. Labirynt jest w postaci kraty prostokątnej o określonym rozmiarze (NxM), w którym niektóre pola (kratki) zajęte są przez kwadratowe bloczki, a niektóre są puste i umożliwiają przejście. Przykład labiryntu o rozmiarze 6x6 wraz z przykładowym sposobem jego reprezentacji za pomocą tablicy dwuwymiarowej znajduje się na rysunku poniżej.



1	1	1	0	0	3
0	1	0	0	2	2
0	0	0	1	0	0
0	4	5	4	4	4
0	4	0	1	0	0
0	4	0	0	0	1

Ogólne wytyczne

Projekt powinien być napisany w języku C z możliwością stosowania podstawowych elementów języka C++ (podobnie jak podczas laboratoriów). Projekt może być pisany w sposób obiektowy, ale całkowicie **zabronione jest użycie biblioteki STL.**

Obsługa plików powinna być zrealizowana za pomocą funkcji z rodziny f????? (fopen, fread itp.) oraz typu FILE. Nie można w tym celu używać mechanizmów C++ (np. fstream).

Za projekt można uzyskać 0-18 punktów. 15 punktów stanowi tzw. 100% pozostałe 3 punkty to wymagania ponadprogramowe ("z gwiazdką").

Realizacja wymagań obowiązkowych konieczna jest aby projekt był w ogóle oceniany.

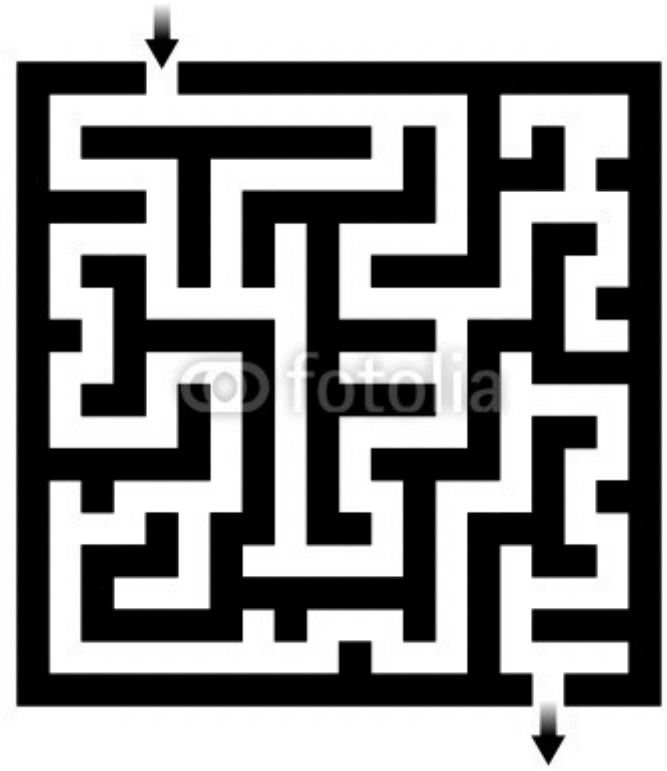
1

Uwaga: W razie niejasności lub niejednoznaczności w poniższym opisie proszę kontaktować sie z autorem instrukcji; pokój 738EA; konsultacje odbywają się w poniedziałki 12:15-14:00

Program powinien być napisany z użyciem szablonu dostępnego w materiałach. Szablon umożliwia uzyskanie zaawansowanych możliwości w zakresie obsługi konsoli w systemie Windows.

Uwaga!. Udostępniona biblioteka działa w systemie Windows i nie ma możliwości łatwego przeniesienia jej do innych systemów. Osoby, które piszą program w systemie Linux powinny wykorzystać bibliotekę **ncurses**.

Podczas oddawania (aby usprawnić oddawanie) należy przygotować labirynt przykładowy. Student oddający powinien móc wczytać go z domyślnego pliku po wybraniu klawisza "i". Labirynt domyślny powinien mieć postać:



(https://de.fotolia.com/id/12318943)

Ściany należy losowo i równomiernie przyporządkować do różnych kategorii (co najmniej 5 kategorii).

Dla uproszczenia problemu rysowania labiryntu w rzucie perspektywicznym, **program i edytor muszą uniemożliwiać tworzenie i obsługę** labiryntów zawierających puste obszary o rozmiarze 2x2 i większe (znacznie utrudnia to poprawne rysowanie w rzucie perspektywicznym).

Pamiętaj żebyś przygotował się do odbioru - koniecznie sprawdź wcześniej czy potrafisz sprawnie uruchomić swój program na komputerze w laboratorium!.

Obsługa programu

Program powinien wykorzystywać klawiaturę w następujący sposób:

strzałki lewo/prawo	zmiana kierunku patrzenia o 90 stopni w wybranym kierunku
strzałka w górę	ruch do przodu
strzałka w dół	ruch do tyłu
d	otwieranie i zamykanie drzwi
i	wczytanie domyślnego labiryntu
q	opuszczenie programu
h	wyświetlenie pomocy
e	wejście do edytora
o	odczytanie labiryntu z pliku o nazwie podanej przez użytkownika
s	zapisanie do pliku o nazwie podanej przez użytkownika
r	restart "gry"
a	aktywacja animacji ruchu do przodu i tyłu

Wymagania obowiązkowe (5 pkt.)

Zaimplementuj wyświetlanie labiryntu w rzucie z góry. Uwzględnij możliwość przemieszczania się po nim i obracania się o 90 stopni za pomocą strzałek.

Przykładowo, w widoku z góry, obrót powoduje tylko zmianę sposobu przedstawienia gracza - wyświetlamy inna strzałkę (plansza jest statyczna a strzałką zaznaczamy kierunek gracza na niej).

Zaimplementuj mechanizm wykrywania kolizji w taki sposób aby uniemożliwić wtargnięcie gracza na ścianę.

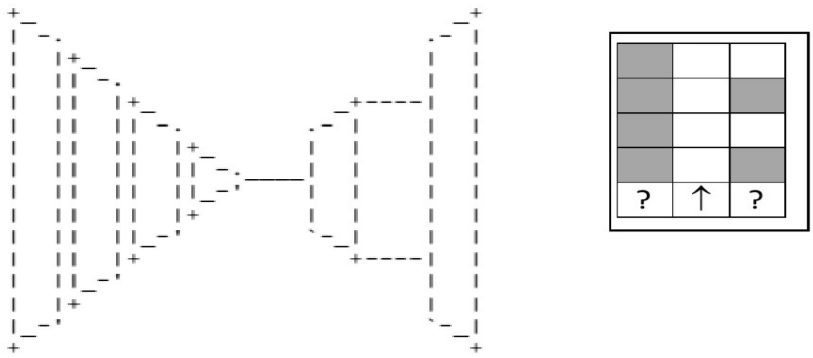
Zapewnij możliwość wczytywania labiryntu z pliku tekstowego o ustalonym przez Ciebie formacie.

Zaplanuj kilka rodzajów ścian o różnym wyglądzie (użycie różnych znaków ASCII do rysowania). Różne typy ścian należy oznaczać wybranymi znakami, np. #@&%$.

Zliczaj i wyświetl czas od wczytania labiryntu oraz liczbę wykonanych ruchów. Wciśnięcie przycisku "r" powinno zresetować liczniki i pozycję gracza do początkowej.

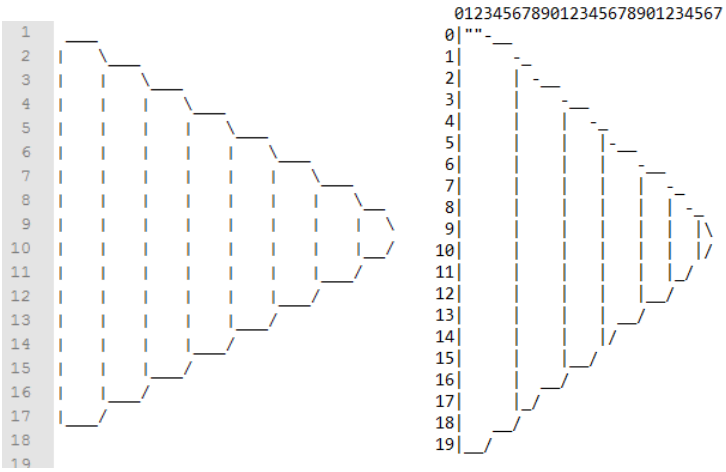
Wymagania nieobowiązkowe (10 pkt.)

(5 pkt.) Zaimplementuj możliwość (równocześnie z rzutem z góry) rysowania labiryntu w rzucie perspektywicznym (tego co widzi gracz). Idea tego rysowania przedstawiona została na rysunku poniżej. Dodatkowo przedstawiono fragment labiryntu w rzucie z góry.



W powyższym przykładzie, położenie gracza i kierunek jego patrzenia zaznaczony jest strzałką. W miejscach znaków zapytania mogą występować dowolne pola (zarówno puste jak również zajęte - nie ma to znaczenia). W tym przypadku, głębokość widzenia wynosi 4. **Twoim zadaniem jest rysować stosując głębokość 7**. Wykorzystaj do tego celu jak największą część ekranu, ale tak aby zmieścić równocześnie podgląd labiryntu w rzucie z góry na tym samym ekranie.

Dłuższe pole widzenia wymaga bardziej łagodnej perspektywy żeby zmieścić je w typowej wielkości konsoli. Na rysunku poniżej widać przykładowy wygląd (tylko jedna strona) w dwóch wariantach.



(1 pkt.) Drzwi - niektóre pola labiryntu to drzwi, które mogą być otwarte lub zamknięte. Pola takie powinny być rysowane w inny sposób niż zwykłe ściany (o ile obraz ściany jest dostatecznie duży; inaczej drzwi otwarte inaczej drzwi zamknięte). Jeżeli stoimy bezpośrednio przed drzwiami i patrzymy na nie na wprost, naciśnięcie klawisza 'd' powinno je otwierać lub zamykać. Otwarcie i zamknięcie powinno być przedstawione animacją. Przez drzwi otwarte można przejść tak jak przez puste pole. Drzwi zamknięte blokują drogę -- tak jak ściana."

(1 pkt.) Zapewnij żeby różne klocki wyglądały w różny sposób (tekstura klocka musi być uwidoczniona w widoku perspektywicznym oraz w widoku z góry w taki sam sposób, żeby ułatwić kontrolę poprawności podczas odbioru).

(2 pkt.) Zaimplementuj edytor poziomów wraz z możliwością zapisu do pliku. Po opuszczeniu labiryntu powinna być możliwa jego natychmiastowa eksploracja bez konieczności wczytywania z

pliku. Edytor powinien uniemożliwiać tworzenie niepoprawnych labiryntów (tj. zawierające puste obszary 2x2).

(1 pkt.) Zaimplementuj opcjonalny tryb (aktywacja i dezaktywacja tego trybu klawiszem a) płynnego przechodzenia do przodu i tyłu (w widoku perspektywicznym). Każdy ruch do przodu lub tyłu o jedno pole wyświetlamy co najmniej w 5 klatkach animacji. Aby "wycisnąć" 5 klatek animacji pierwsza ściana powinna mieć szerokość co najmniej 5. Pozostałe mogą mieć mniejszą szerokość.

Wymagania z gwiazdką

Możesz wybrać tylko jeden z wariantów. Wariant A umożliwia zdobycie 1 punktu dodatkowego, natomiast wariant B umożliwia zdobycie 3 punktów.

Wariant A

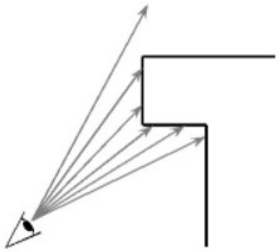
(1 pkt.) Rysuj na podłodze strzałkę prowadzącą najkrótszą drogą do wyjścia z labiryntu. Powinno działać dla każdego labiryntu - w szczególności tych utworzonych w edytorze. Droga do wyjścia powinna wyliczać się automatycznie. W programie powinna się również wyświetlać odległość do wyjścia. Jednostką jest oczywiście [1 ruch], lub inaczej pole.

Wariant B

(3 pkt.) Zaimplementuj możliwość płynnej zmiany kąta patrzenia. Oraz umożliwiaj płynne poruszanie się po labiryncie.

Szkic rozwiązania:

Pomocne jest w tym przypadku wyznaczanie długości promieni rzutowanych od obserwatora (pod różnymi kątami względem jego kąta patrzenia).



Im dłuższy promień (zanim trafi na ścianę) tym niższa wysokość ściany widziana przez obserwatora. Zatem dla każdej kolumny ekranu należy wyznaczyć długość promienia i narysować pionowy prążek o wysokości odwrotnie proporcjonalnej do tej długości.

Płynne poruszanie się po labiryncie stanowi naturalne dopełnienie płynnej zmiany kąta patrzenia w tym sensie, że dobre rozwiązanie problemu obliczania długości promieni powinno dobrze zaadaptować się dla problemu płynnego poruszania się po labiryncie.

Uwagi dodatkowe:

Jeden z waszych kolegów zapytał:

W instrukcji dla pierwszego projektu napotkałem się na pewną niejasność dotyczącą punktacji. Czy można wykonać resztę punktów niewymaganych jeśli nie zaimplementowano perspektywy pierwszej osoby? Niektóre z nich są bezpośrednio związane z perspektywą, więc te nie są możliwe, lecz na przykład drzwi można zaimplementować bez niej, aczkolwiek nie spełniłoby to wymagań dotyczących animacji. Podobnie jest z różnymi rodzajami ścian. Z kolei edytor poziomów w ogóle nie potrzebuje FPP aby spełniał wymagania w 100%.

Odpowiedź:

Edytor poziomów rzeczywiście nie wymaga FPP. Jak się porządnie zaimplementuje edytor bez FPP można liczyć na 2pkt.

Drzwi wymagają FPP w związku z animacją (jak sam Pan zauważył), która jest częścią tego wymagania. Realizacja bez animacji może być uznana częściowo lub wcale (w zależności od oceny prowadzącego).

Rożne rodzaje ścian cytuję: "tekstura klocka musi być uwidoczniona w widoku perspektywicznym oraz w widoku z góry". Teksturowanie w widoku z góry jest ujęte w wymaganiach obowiązkowych, cytuję: "Różne typy ścian należy oznaczać wybranymi znakami, np. #@&%$.". Tutaj nie można liczyć na dodatkowe punkty.