

Podstawy programowania, Czwartek 11.05.2015, 16:15-17:45

Projekt, część 3

1. Zadanie

Projekt polega na stworzeniu logicznej gry komputerowej działającej w trybie tekstowym o nazwie „Minefield”.



2. Cele

Celem projektu jest napisanie bardziej rozbudowanego programu niż było to możliwe na pojedynczych zajęciach i przeciwienie wcześniej nauczonych mechanizmów języka C. Ponieważ jest to projekt, wymagana jest również własna kreatywność oraz inwencja – nie wszystko w treści zadania będzie napisane krok po kroku jak to zrobić.

Uwaga! Zalecane jest szerokie stosowanie zewnętrznych funkcji – w dobrym projekcie funkcja `main` powinna być jak najkrótsza.

3. Sprawdzanie min

Zadaniem z ostatnich zajęć było stworzenie planszy, na której rozmieściliśmy miny oraz saperą. Dzisiaj dokończymy grę dając saperowi umiejętność wykrywania min. Celem samej gry jest przejście saperą z lewego dolnego rogu planszy do prawego górnego rogu planszy, omijając miny. Wejście na minę oznacza przegraną i kończy grę. W jaki sposób zatem saper sprawdza miny? Saper ma możliwość sprawdzenia jedynie najbliższych pól. Zauważmy, że punkt, w którym saper się znajduje ma 8 pól sąsiadujących. Zatem dla każdej pozycji saperą (po każdym jego ruchu) sprawdzamy, czy któraś z min nie znajduje się na jednym z 8 pól sąsiadnych – jeśli nie znajduje się, na ekranie wypisywany jest tekst „Brak min!”, jeśli zaś na jednym z 8 pól sąsiadujących znajduje się jedna lub więcej min, wypisywany jest tekst „Miny!”.

W naszym projekcie miny są widoczne na ekranie, ale nie stanowi to żadnego problemu, aby dla finalnego gracza wyłączyć ich rysowanie na planszy. Zadaniem gracza jest zatem przejście z jednego do drugiego rogu planszy omijając miny na podstawie informacji posiadanej przez saperą (zauważmy, że gracz nie wie, na którym z 8 pól sąsiadujących mina się znajduje!). Wprowadzona na ostatnich zajęciach „ścieżka bezpieczeństwa” pozwala graczowi na bezpieczne cofnięcie się na odwiedzone wcześniej, bezpieczne (bez min), pola.

Przykład informacji o minach jest pokazany na obrazkach poniżej.

2) Wejście na minę:

Tworzymy funkcję analogiczną do funkcji powyższej (zwracającą `bool`), która z kolei sprawdza, czy pozycja `x,y` sapera zgadza się z którąś z min z tablicy `min` (znowu iterujemy po tablicy `min` i dla każdejminy sprawdzamy czy `x` oraz `y` zgadzają się z `saperem`). Jeśli zgadza się, funkcja zwraca `true`, jeśli nie, zwraca `false`. Funkcję wykorzystujemy w pętli nieskończonej `while` odpowiadającej za ruch. Jeśli funkcja zwraca `false`, to nic się nie dzieje, jeśli zwraca `true`, to przerywamy pętlę `while` i wypisujemy odpowiedni komunikat o przegranej na ekran. Gra wraca do menu głównego.

3) Wygrana:

Funkcja działa identycznie jak wejście na minę, z tą różnicą, że sprawdzamy tylko czy pozycja sapera nie jest równa prawemu górnemu punktowi planszy. Jeśli tak, funkcja zwraca `true` i przerywa pętlę `while` wypisując odpowiedni komunikat na ekran. Dodatkowo (po przerwaniu pętli) program pyta się użytkownika, czy zapisać wynik do pliku oraz prosi o podanie nazwy użytkownika. Na koniec gra wraca do menu głównego.

Punktacja:

1. Stworzenie ścieżki bezpieczeństwa (i działający program z poprzednich zajęć) **(1 p.)**
2. Obsługa sprawdzania min na 8 sąsiednich polach **(3 p.)**
3. Obsługa sytuacji przegranej – wejścia na minę **(2 p.)**
4. Obsługa sytuacji wygranej (prawy górny róg planszy), zliczenie liczby ruchów oraz zapisanie wyniku do pliku **(3 p.)**

Zrozumienie, analiza treści, projekt programu + poprawność i estetyka kodu. Punkty za ten element można dostać mając skończoną przynajmniej połowę 1 części programu **(1 p.)**