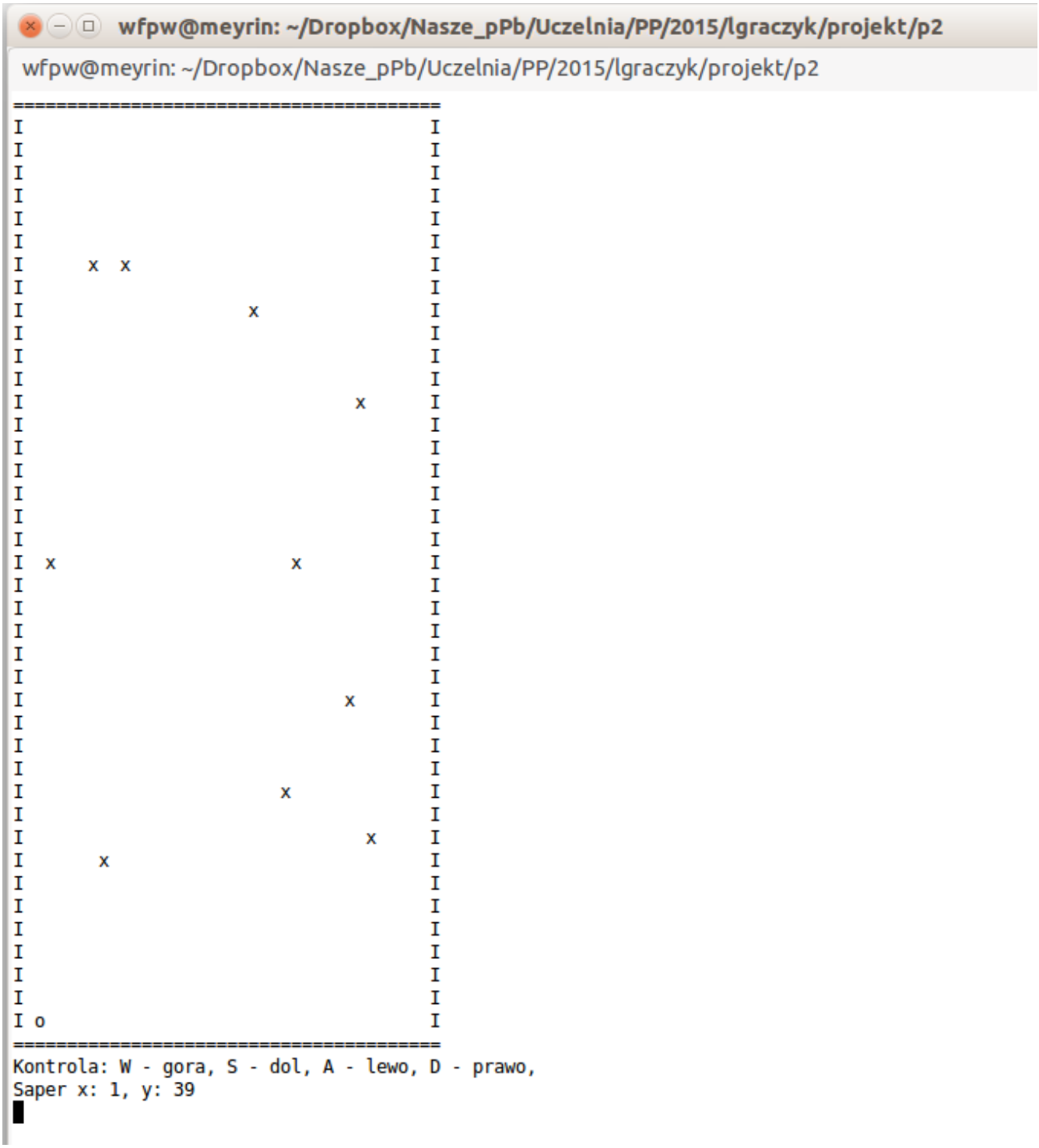
Podstawy programowania, Czwartek 28.05.2015, 16:15-17:45
Projekt, część 2

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 przećwiczenie 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. Plansza
W zeszłym tygodniu został stworzony szkielet gry w postaci menu głównego oraz możliwości wyboru nowej gry, w której to występowało określenie liczby min oraz losowanie ich pozycji (wartości x i y od 1 do 39). W drugiej części projektu głównym zadaniem jest stworzenie planszy, w której rozmieszczone zostaną wcześniej stworzone miny oraz saper. Saper powinien mieć możliwość poruszania się przy pomocy klawiszy „W”, „S”, „A”, oraz „D”. Saper powinien być oznaczony na planszy za pomocą literki „o”, zaś miny za pomocą literki „x”. W programie należy również zaznaczyć tzw. „ścieżkę bezpieczeństwa”, czyli drogę, którą saper pokonał od początku gry.
Przykładowa plansza przedstawiona jest na poniższym rysunku:



Szczegóły implementacji

Na 1 części laboratorium stworzona została funkcja `void NowaGra()`, w której to funkcji tworzone było pole minowe poprzez funkcję `UstawMiny`. Na dzisiejszych zajęciach do funkcji `NowaGra` dodajemy planszę oraz ruch sapera.

1) Narysowanie planszy:

W celu jej stworzenia najprościej napisać funkcję `void plansza(Punkt s, PoleMinowe m)`, wywoływaną w funkcji `NowaGra`, która przyjmuje za argumenty pozycję sapera `s` (saper jest

obiektem typu `Punkt`) oraz pole minowe (obiekt typu `PoleMinowe`) `m`. Zakładamy, że plansza jest stałego rozmiaru 40x40 (wartość tą można zdefiniować poprzez stałą globalną `#define PLANSZA 40`). Przedstawiona powyżej plansza jest wynikiem zastosowania odpowiedniej kombinacji funkcji `printf` umieszczonych wewnątrz dwóch zagnieżdżonych pętli `for`. Dla każdego punktu (x,y) planszy musimy sprawdzić, czy odpowiada on współrzędnym położenia sapera oraz min (każdej miny z tablicy min z obiektu `PoleMinowe m`). Jeśli dany punkt planszy nie odpowiada żadnemu z powyższych, wypisujemy na ekran spację, jeśli zaś odpowiada, to wypisujemy „o" dla sapera lub „x" dla miny w danym punkcie. Sapera domyślnie (na początku gry) umieszczamy w lewym dolnym rogu planszy.

2) Ruch sapera:

Ruch sapera polega na zmianie położeń (czyli wartości x oraz y) sapera `s` (obiektu typu `Punkt`) oraz przerysowywaniu planszy po każdej takiej zmianie. W tym celu należy w funkcji `NowaGra` zaimplementować nieskończoną pętlę, w której każda zmiana pozycji będzie powodować czyszczenie ekranu terminala (komenda `system("clear")` znana z części 1 projektu) oraz ponowne narysowanie planszy poprzez wywołanie funkcji `plansza`. Jedyną niewiadomą pozostaje zmiana pozycji sapera – jak to zrobić? Do tego celu należy wykorzystać funkcję `char getch()` z pliku `getchar.h` (jest to podobna funkcja do `scanf`, z tą różnicą, że nie wymaga potwierdzenia wprowadzenia znaku poprzez wciśnięcie klawisza Enter).

Przykład wykorzystania funkcji `char getch()` - wczytanie pojedynczego znaku z klawiatury i jego wypisanie:

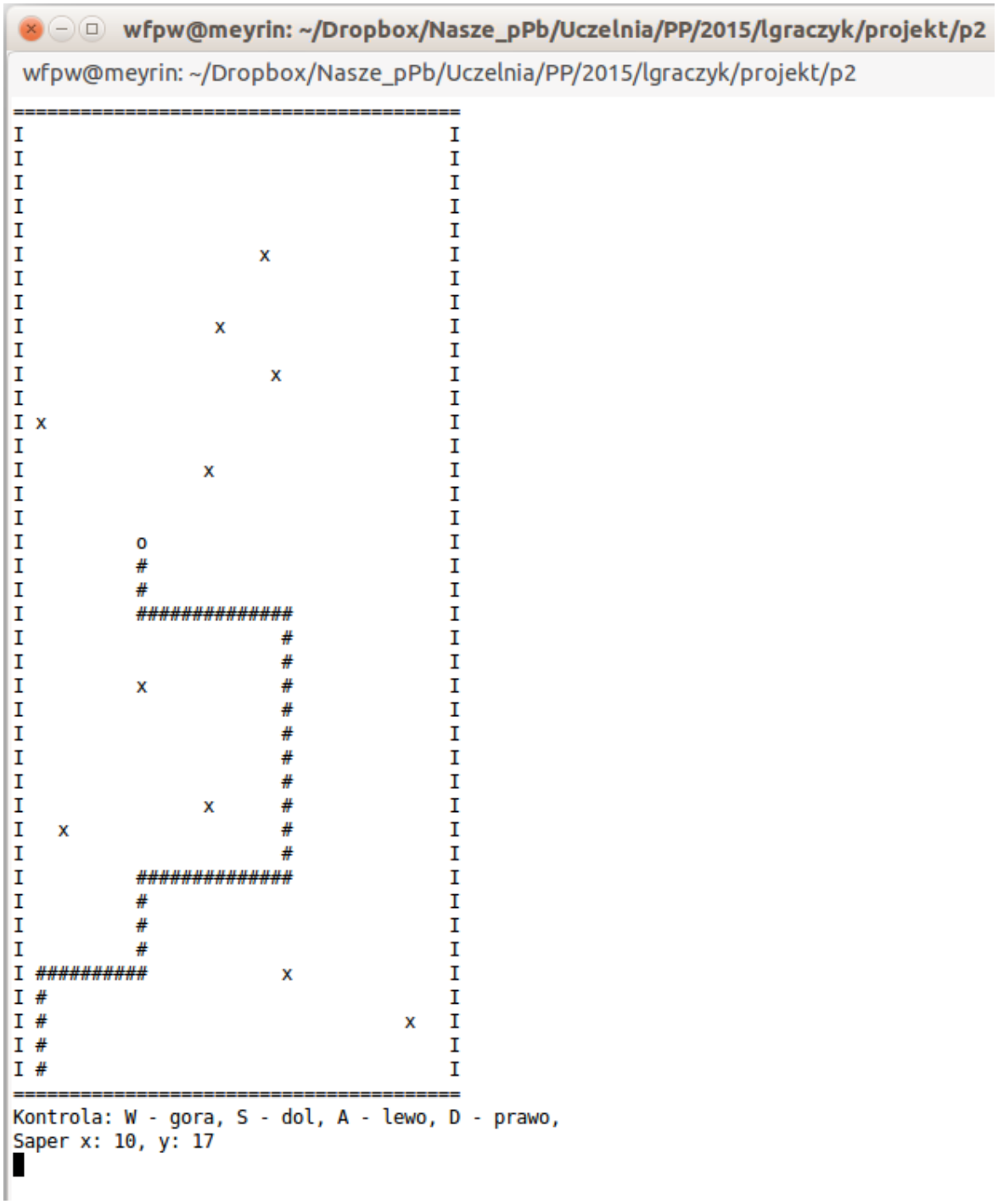
```
#include <stdio.h>
#include "getchar.h"

int main(void)
{
  char znak;
  znak = getch();
  printf("Podany znak to: %c\n",znak);

  return 1;
}
```

W przypadku ruchu sapera, wciśnięcie klawiszy:
„w" - przesuwa sapera o 1 w górę (czyli wartość y położenia sapera zmniejsza się o 1)
„s" - przesuwa sapera o 1 w dół (czyli wartość y położenia sapera zwiększa się o 1)
„a" - przesuwa sapera o 1 w lewo (czyli wartość x położenia sapera zmniejsza się o 1)
„d" - przesuwa sapera o 1 w prawo (czyli wartość x położenia sapera zwiększa się o 1)
„0" - wychodzi z gry (przerywa pętlę przerysowującą planszę) i wraca do menu głównego

3) „Ścieżka bezpieczeństwa":

Ostatnim elementem zadania jest narysowanie tzw. „ścieżki bezpieczeństwa", czyli zaznaczenie drogi, którą przebył saper od momentu uruchomienia gry. Aby zaimplementować ten element najlepiej jest stworzyć dwuwymiarową tablicę typu `bool` o rozmiarze `PLANSZA` (wcześniej zdefiniowanej zmiennej globalnej). Pola tablicy o współrzędnych (x,y) domyślnie ustawione są na `false` (nie odwiedzone przez sapera); natomiast jeśli saper pod wpływem naszego ruchu dane pole odwiedzi, jego wartość w tablicy ustawiana jest na `true`. W funkcji rysującej plansze musi istnieć warunek sprawdzający stan danego elementu tablicy: jeśli `true`, to wypisywany jest znak „#", jeśli `false`, to spacja.

Przykład planszy po przejściu przez sapera pewnego obszaru:



Punktacja:

1. Stworzenie pustej planszy (**1 p.**)
2. Narysowanie min wewnątrz planszy (**2 p.**)
3. Stworzenie sapera oraz narysowanie go w punkcie początkowym (**1 p.**)
4. Obsłużenie ruchu sapera za pomocą klawiszy „W”, „S”, „A”, „D”. (**3 p.**)
5. Implementacja „ścieżki bezpieczeństwa”. (**2 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.**)