Politechnika Śląska

Wydział Matematyk Stosowanej

Kierunek Informatyka

Gliwice, 01.10.2020

Programowanie I

**projekt zaliczeniowy**

**"*The Snake Game*"**

**Szymon Jakubiec gr. lab. 1/1**

**1. Opis projektu.**

*Moim projektem jest gra typu Snake.*

**2. Wymagania**

*- możliwość wyboru jak szybko porusza się wąż*

*- granie w klasycznego Snake’a , czyli zjadanie jabłek i niedobijanie do siebie lub ścian*

*- rozgrywka w kolorowej grafice przypominającej węża pełzającego po trawie otoczonego krzakami*

*- pokazywanie w trakcie gry ilości zdobytych punktów, którymi będą zjedzone jabłka*

**3. Przebieg realizacji**

*Mój projekt znajduje się w całości w jednym pliku – „Główny.cpp”. Plik „testowy.cpp” jest kopią, zapisem pewnego stanu pliku głównego w razie jakby coś nie wyszło.*

*Opis algorytmu:*

*Po wyświetleniu wstępu i opcji, o których piszę w następnej części program czyści ekran i dopasowuje kolory. Następnie wypisuje po kolei: pustą planszę, licznik punktów i wylosowaną pozycję jabłka. Domyślnie wąż zaczyna poruszać się w prawo, toteż jest ustawiony nieco z lewej. Właściwa gra toczy się w pętli. Pól na planszy jest 150 z czego jedno z nich zajmuje już głowa węża, więc mamy do zjedzenia 149 jabłek.*

*Pętlę zaczynamy wpisaniem ‘O’ – głowy węża oraz, oraz wpisaniem ‘o’ jako części jego ciała w poprzednim miejscu głowy, jeśli wąż ma chociaż długość równą 1(głowa ma długość 0).*

*Następnie program zostaje zatrzymany na tak długo jak to wybraliśmy w ustawieniach. Czekamy przy tym czy jakiś klawisz zostanie naciśnięty. Jeśli to będzie jeden z wsad to zmieniamy kierunek ruchu, przy czym program został wyczulony na to by wąż mógł skręcić tylko o 90 stopni np. po poruszaniu się w prawo mógł tylko skręcić w górę lub dół itd.*

*Dalej algorytm pyta czy jabłko zostało wcześniej zjedzone. Jeśli nie (am == false) to przesuwamy „ciało” węża bez głowy i usuwamy go z dotychczas ostatniej pozycji.*

*Obraz zawierający tekst

Opis wygenerowany automatycznie*

*A jeśli zjadł (am == true) to zmieniamy am na false, wydłużamy węża w miejscu zajmowanym przez jego dotychczasowy koniec, dodajemy punkt w liczniku pod planszą, losujemy nową pozycję dla jabłka tak długo aż znajdziemy puste pole, a reszta pozostaje tak samo.*

*Na końcu sprawdzamy, w którą stronę poruszy się głowa węża i co się może stać. Jeśli trafi na puste pole to prostu się tam przemieszcza, jeśli trafi na jabłko to je zjada (am = true), a jak do czegoś dobije to gra się przerywa.*

**4. Instrukcja użytkownika**

*Na początku program tworzy wstęp gdzie wypisuje tytuł, autora i polecenie naciśnięcia klawisza ENTER, by przejść dalej.*

*Obraz zawierający tekst, monitor, zrzut ekranu, wyświetlanie

Opis wygenerowany automatycznie*

*Dalej przechodzimy do opcji, gdzie możemy wybrać prędkość z jaką ma poruszać się wąż, czyli na jak długo program się zatrzymuje przy każdym ruchu. (wolny – 200ms, średni – 150ms, szybki – 100ms). Potwierdzamy swój wybór klawiszem ENTER i przechodzimy do rozgrywki.*

*Obraz zawierający tekst, zrzut ekranu, monitor, sprzęt elektroniczny

Opis wygenerowany automatycznie*

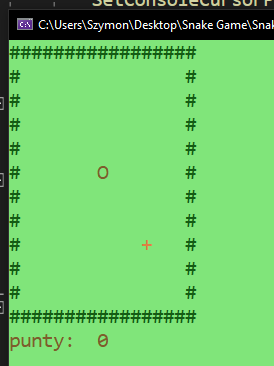
*W trakcie gry poruszamy się używając klawiszy:*

*w - górę,*

*s - dół,*

*a – w lewo,*

*d - prawo.*

*Rozgrywkę urozmaiciłem zmieniając kolory tła i wypisywanego tekstu tak aby przypominało to węża pełzającego po trawie otoczonego krzakami. Poniżej planszy wypisywane są zdobyte punkty.*

**5. Podsumowanie i wnioski.**

*Zrealizowałem:*

*- kolorowe menu dostosowujące się do szerokości ekranu konsoli*

*- opcję dostosowującą się do szerokości ekranu konsoli*

*- działająca podstawa algorytm do Snake’a w kolorze*

*- zliczanie punktów i pokazywanie ich po końcu rozgrywki*

*Miałem problemy z:*

*- działaniem podstawy algorytmu:*

*\* przepisywaniem współrzędnych między „częściami” węża, by mógł się poruszać*

*\* momentem, w którym wąż miał zjeść jabłko*

*- funkcją na losowanie miejsca na nowe jabłko*

*Nie udało mi się zrealizować:*

*- planszy i końcowego komunikatu dostosowującego się do szerokości ekranu konsoli*

*- opcji ustawienia swoich klawiszy do sterowania*