|  |  |  |
| --- | --- | --- |
| **Uniwersytet Rzeszowski Wydział Matematyczno-Przyrodniczy  Informatyka** | | |
| Przedmiot: **Systemy Operacyjne 2** – Laboratorium | | |
| Nazwisko i imię Mateusz Baczek | Laboratorium nr 1  Temat:  **Projekt aplikacji „Snake” z wykorzystaniem możliwości powłoki systemowej Bash** | Data wykonania:  14.12.2019 |
| Grupa laboratoryjna nr 1 | Data oddania:  26.10.2019 |
| Ocena: |

Spis treści

**Nie znaleziono żadnych pozycji spisu treści.**

# Założenia projektu

## Założenia ogólne

Stworzenie aplikacji ma na celu demonstrację możliwości powłoki systemowej Bash jako platformy do tworzenia podstawowych skryptów z zachowaniem podobnym do stylu aplikacji typu okienkowego – ograniczenie użytkownika wyłącznie do możliwości oferowanych przez skrypt. W tym przypadku użytkownik może operować w programie poprzez klawisze **W, A, S, D** oraz **Q**.

## Wykorzystane komendy

|  |  |
| --- | --- |
| **Komendy** | |
| **printf** – polecenie zapewnia metodę wyświetlania w konsoli wstępnie sformatowanego tekstu podobnie do interfejsu dla języka C | |
| printf "\e[?25l" | pozwala na wyłączenie wyświetlania kursora(podczas startu skryptu) |
| printf "\e[?25h" | pozwala na włączenie wyświetlania kursora(po zakończeniu działania) |
| printf "\e[3;5HX" | przesuwa kursor na 3-cią linię, 5 pozycję wewnątrz konsoli i wypisuje „X” niezależnie od wyświetlanego stanu konsoli (mogą znajdować się również inne wykonane polecenia) |
| printf $body | grep –o ‘[0-3]$’ | wyświetlenie pierwszej wartości ciągu znaków (ciała) zawierającej znaki 0, 1, 2 lub 3 |
| **stty** – wypisanie lub zmiana ustawień terminala | |
| stty -echo | wyświetlanie każdego znaku z wejścia (start skryptu) |
| stty echo | zakończenie działania powyższego polecenia (przywraca poprzedni tryb) |
| **kill** – program wbrew nazwie pozwalający na przesyłanie sygnałów do procesów pracujących w systemie operacyjnym oraz poprawne zamknięcie procesu z zachowaniem danych | |
| kill SIGHUP 850 | przesłanie do procesu sygnału o zmianie kierunku ruchu (góra) |
| kill SIGINT 850 | przesłanie do procesu sygnału o zmianie kierunku ruchu (prawo) |
| kill SIGABRT 850 | przesłanie do procesu sygnału o zmianie kierunku ruchu (dół) |
| kill SIGILL 850 | przesłanie do procesu sygnału o zmianie kierunku ruchu (lewo) |
| kill SIGQUIT 850 | przesłanie do procesu sygnału o wymuszeniu zakończenia działania skryptu „Hangup” |
| **eval** – konstruuje polecenie, łącząc argumenty ze sobą. Skonstruowane polecenie zostaje odczytane i wykonane przez powłokę. | |
| eval printf \"\${board$bh[\*]}\" | polecenie wyświetla wszystkie elementy tablicy w jednej linii poprzez zastosowanie znacznika **[\*]**, wykorzystanie znacznika **[@]** wyświetla wszystkie elementy w osobnej linii |
| eval "board$bh[$bw]=' '" | przypisuje wartość **‘ ‘** dla elementu tablicy **board** w położeniu bh, bw |
| eval "pos=\${board$food\_row\_pos  [$food\_col\_pos]}" | do zmiennej **pos** przypisuje wartość z tablicy **board** w położeniu food\_row\_pos, food\_col\_pos |
| eval "board$new\_row\_pos  [$new\_col\_pos]=\"$body\_sign\"" | przypisuje zmienną typu łańcuchowego **body\_sign** do elementu tablicy **board** w pozycji new\_row\_pos, new\_col\_pos |
| **read** – służy do wczytywania danych ze standardowego wejścia, którym może być klawiatura i i umieszcza je w zmiennych | |
| read -sn1 key | **-s** – nie wyświetla wczytywanych znaków **-n** - wczytuje podaną ilość znaków (w tym przypadku **1**)  przypisuje do zmiennej **key** |
| **trap** – przechwytuje sygnały (przerwy programowe) wspierane przez mechanizmy komunikacji IPC | |
| trap "move\_av=0;" SIGHUP | przechwytuje sygnał **SIGHUP** i zmiennej **move\_av** przypisuje wartość liczbową **0** |
| **Pozostałe** | |
| IFS='' – [Internal Field Separator] znak wyświetlany pomiędzy zmienną za znakiem **$** a kolejną zmienną. Domyślna wartość wynosi <space><tab><newline> | |
| declare body\_sign="o" – deklaracja łańcucha znaków o wartości **o** | |
| sleep $speed – opóźnia wykonanie dalszych fragmentów kodu | |

## Zasady gry

Gracz za pomocą klawiszy **W, A, S, D** steruje obiektem złożonym ze znaków zadeklarowanych na samym początku działania skryptu – domyślnie „o”, który jak sama nazwa wskazuje jest odpowiednikiem węża. Wąż ma za zadanie zebranie jak największej ilości innych obiektów symbolizujących pożywienie – znak „@”. Gracz ma zadanie unikanie własnego ogona/reszty ciała oraz obrysu pola gry „X”. Zakończenie gry odbywa się w 3 przypadkach

- wymuszenie zatrzymania gry poprzez wciśnięcie klawisza „q”/”Q”

- wejście obiektu w obrys pola gry

- wejście we własne ciało

# Struktura skryptu

## Deklaracja zmiennych

## Inicjalizacja skryptu

## Inicjalizacja widoku

## Zmiana wartości planszy – symulacja ruchu

## Generowanie obiektów na planszy

## Komunikacja między procesami

## Przechwytywanie akcji na klawiaturze

## Zakończenie działania procesów

# SIGQUIT - wyjscie

# SIGHUP - gora

# SIGINT - prawo

# SIGILL - lewo

# SIGABRT - dol

# SIGHWINCH - stop