Gra w ‘Kółko i krzyżyk’

Dokumentacja

Albert Winiarski

„Aplikacja z grą w kółko i krzyżyk w terminalu, dla dwóch osób.”

Fortran | Semestr letni | 2024

**Informatyka Stosowana - 2 rok**

* **Wstęp**
  + W celu wykonania zadania polegającego na stworzeniu krótkiego projektu, postanowiłem napisać program, który umożliwi grę w kółko i krzyżyk w terminalu.
  + Docelowo gra miała pozwalać na grę dla dwóch osób, dzięki technice zwanej „hot seat”, gdzie jeden użytkownik gra na zmianę z drugim.
* **Generalny opis działania gry**
  + Gra “**Kółko i Krzyżyk**” (Tic-Tac-Toe) to prosta gra dwuosobowa, w której gracze na przemian stawiają swoje znaki (X lub O) na planszy 3x3.
  + Celem gry jest ustawienie trzech swoich znaków w linii poziomej, pionowej lub ukośnej. Wtedy rozgrywka się kończy.
  + Cały program został napisany w języku Fortran i jest przeznaczony do uruchamiania w terminalu. Tam przebiega cała rozgrywka.
* **Wymagania systemowe**
  + Komputer z zainstalowanym kompilatorem Fortran (np. GNU Fortran - gfortran)
  + System operacyjny: Linux, macOS lub Windows.
* **Struktura kodu**
  + Kod programu składa się z następujących sekcji:
    - Deklaracje zmiennych
    - Inicjalizacja planszy
    - Główna pętla gry
    - Podprogramy i funkcje:
      * **initialize\_board**: Inicjalizuje planszę pustymi polami
      * **print\_board**: Wyświetla aktualny stan planszy
      * **check\_winner**: Sprawdza, czy dany gracz wygrał
      * **menu**: Wyświetla menu główne
      * **get\_move**: Pobiera ruch gracza
      * **save\_result**: Zapisuje wynik gry do pliku
      * **instructions**: Wyświetla instrukcje gry
* **Implementacja:**
  + **Deklaracje zmiennych:**
    - Program deklaruje główne zmienne, takie jak:
      * **board** (plansza gry),
      * **row**, **col** (wprowadzone przez użytkownika rząd i kolumna),
      * **turn** (licznik tur),
      * **game\_over** (flaga końca gry),
      * **valid\_move** (flaga poprawności ruchu) oraz
      * **player** (aktualny gracz).
  + **Inicjalizacja planszy**
    - Podprogram **initialize\_board** ustawia wszystkie pola planszy na puste (' '):
  + **Główna pętla gry**
    - Pętla DO WHILE kontroluje przebieg gry:
      * Gracz wprowadza rząd i kolumnę
      * Program sprawdza poprawność ruchu
      * Aktualizuje planszę
      * Sprawdza warunki wygranej lub remisu
      * Zmienia gracza po każdym poprawnym ruchu
* **Poszczególne funkcje i podprogramy**
  + **initialize\_board:**
    - SUBROUTINE initialize\_board(b)
    - CHARACTER(1), INTENT(OUT) :: b(3,3)
    - INTEGER :: i
    - DO i = 1, 3
    - b(i, 1) = ' '
    - b(i, 2) = ' '
    - b(i, 3) = ' '
    - END DO
    - END SUBROUTINE initialize\_board
  + **print\_board -** wyświetla aktualny stan planszy**:**
    - SUBROUTINE print\_board(b)
    - CHARACTER(1), INTENT(IN) :: b(3,3)
    - INTEGER :: i
    - PRINT \*, ' 1 2 3'
    - DO i = 1, 3
    - WRITE(\*,'(I1,2X,A,1X,A,1X,A)') i, b(i,1), b(i,2), b(i,3)
    - END DO
    - END SUBROUTINE print\_board
  + **check\_winner –** funkcja sprawdza, czy dany gracz wygrał**:**
    - FUNCTION check\_winner(b, p) RESULT(win)
    - CHARACTER(1), INTENT(IN) :: b(3,3)
    - CHARACTER(1), INTENT(IN) :: p
    - LOGICAL :: win
    - INTEGER :: i
    - win = .FALSE.
    - DO i = 1, 3
    - IF (ALL(b(i,:) == p) .OR. ALL(b(:,i) == p)) THEN
    - win = .TRUE.
    - RETURN
    - END IF
    - END DO
    - IF (b(1,1) == p .AND. b(2,2) == p .AND. b(3,3) == p) THEN
    - win = .TRUE.
    - ELSE IF (b(1,3) == p .AND. b(2,2) == p .AND. b(3,1) == p) THEN
    - win = .TRUE.
    - END IF
    - END FUNCTION check\_winner
  + **menu** – podprogram wyświetla menu główne:
    - SUBROUTINE menu(choice)
    - INTEGER, INTENT(OUT) :: choice
    - INTEGER :: choice\_local
    - PRINT \*, 'Witaj w grze Kółko i Krzyżyk!'
    - PRINT \*, '1. Nowa gra'
    - PRINT \*, '2. Instrukcje'
    - PRINT \*, 'Wybierz opcję (1-2):'
    - READ \*, choice\_local
    - choice = choice\_local
    - END SUBROUTINE menu
  + **get\_move** – podprogram pobiera ruch gracza:
    - SUBROUTINE get\_move(row, col, b)
    - INTEGER, INTENT(OUT) :: row, col
    - CHARACTER(1), INTENT(IN) :: b(3,3)
    - LOGICAL :: valid
    - valid = .FALSE.
    - DO WHILE (.NOT. valid)
    - PRINT \*, 'Podaj rząd i kolumnę (1-3):'
    - READ \*, row, col
    - IF (row >= 1 .AND. row <= 3 .AND. col >= 1 .AND. col <= 3) THEN
    - IF (b(row, col) == ' ') THEN
    - valid = .TRUE.
    - ELSE
    - PRINT \*, 'To pole jest już zajęte, spróbuj ponownie.'
    - END IF
    - ELSE
    - PRINT \*, 'Nieprawidłowe dane, spróbuj ponownie.'
    - END IF
    - END DO
    - END SUBROUTINE get\_move
  + **save\_result** – podprogram zapisuje wynik gry do pliku:
    - SUBROUTINE save\_result(winner)
    - CHARACTER(1), INTENT(IN) :: winner
    - CHARACTER(LEN=20) :: filename
    - INTEGER :: unit, iostat
    - filename = 'wyniki\_gry.txt'
    - OPEN(NEWUNIT=unit, FILE=filename, STATUS='UNKNOWN', ACTION='WRITE', IOSTAT=iostat)
    - IF (iostat /= 0) THEN
    - PRINT \*, 'Nie można otworzyć pliku wyników.'
    - RETURN
    - END IF
    - IF (winner == ' ') THEN
    - WRITE(unit, \*) 'Gra zakończona remisem.'
    - ELSE
    - WRITE(unit, \*) 'Gracz ', winner, ' wygrał grę.'
    - END IF
    - CLOSE(unit)
    - END SUBROUTINE save\_result
  + **instructions** – podprogram wyświetla instrukcje gry:
    - SUBROUTINE instructions()
    - PRINT \*, 'Instrukcje gry Kółko i Krzyżyk:'
    - PRINT \*, '1. Gra odbywa się na planszy 3x3.'
    - PRINT \*, '2. Gracze na przemian stawiają swoje znaki (X lub O).'
    - PRINT \*, '3. Celem jest ustawienie trzech swoich znaków w linii poziomej, pionowej lub ukośnej.'
    - PRINT \*, '4. Gracz, który pierwszy ustawi trzy znaki w linii, wygrywa.'
    - PRINT \*, '5. Jeśli wszystkie pola są wypełnione i żaden z graczy nie ustawi trzech znaków w linii, gra kończy się remisem.'
    - END SUBROUTINE instructions
* **Przykłady uruchomienia** 
  + Po uruchomieniu programu, użytkownik wybiera, czy chce rozpocząć nową grę, czy zobaczyć instrukcję.
  + Jeżeli wybierze opcję nowej gry, to zobaczy planszę 3x3 z numerami kolumn i rzędów. Gracze na przemian wprowadzają swoje ruchy, podając rząd i kolumnę.
  + Przykład rozpoczęcia rozgrywki:

**Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie**

* **Podsumowanie:**
  + Program jest prostą implementacją gry “ Kółko i Krzyżyk” w języku Fortran.
  + Program ma spory potencjał do rozszerzenia o nowe dodatkowe funkcje w przyszłości.