Juliusz Stańczyk

107408, D

Zadanie 2, gra Gomoku

# Oświadczenie

Świadomy odpowiedzialności prawnej oświadczam, że niniejszy projekt został napisany przeze mnie samodzielnie i nie zawiera treści uzyskanych w sposób niezgodny z obowiązującymi przepisami.

Oświadczam również, że przedstawiona praca nie była wcześniej przedmiotem procedur związanych z uzyskaniem tytułu zawodowego lub zaliczenia przedmiotu z żadnej uczelni ani szkoły.

# Manual

## Zadanie, które program ma realizować

Program ma za zadanie symulować grę w Gomoku. Na planszy 15x15 gracze na zmianę stawiają kółko i krzyżyk(na wolnych polach). Wygrywa ten który ułoży pięć swoich znaków w pionie, poziomie lub ukosie. Gra realizuje możliwość zagrania z komputerem, odmianę gry swap na dwie osoby oraz zapis i wczytanie stanu gry.

## Lista opcji do wyboru z krótkim opisem każdej z nich

1. Na początku program prosi nas o wybranie trybu gry z drugim graczem lub komputerem. Poprawne dane: 1 lub 2. W przypadku wpisania jakiejkolwiek litery, program wyświetla komunikat o błędzie i prosi o ponowne wprowadzenie poprawnych danych. W przypadku wpisania innej cyfry powtarza całą procedurę czyli wcześniejszą prośbę o wybór oznacza to, że program jest przygotowany na błędne dane .
2. Następnie gra prosi nas o wpisane nazwy pliku w którym będzie zapisywać aktualny stan gry. Poprawne dane: wszystko. Program nie sprawdza czy nazwa zakończona jest „.txt” lub innym rozszerzeniem. W przypadku nieotwarcia pliku program kontynuuje poprawne działanie. Oznacza to, że program nie kontroluje wprowadzonych danych.
3. Następnie program prosi o informacje czy wczytać wcześniejszy stan gry. Poprawne dane: 1 lub 2. W przypadku wpisania jakiejkolwiek litery program wyświetla komunikat o błędzie, prosi o ponowne wprowadzenie poprawnych danych. W przypadku wpisania innej cyfry powtarza całą procedurę czyli wcześniejszą prośbę o wybór. Oznacza to, że program jest przygotowany na błędne dane.
   1. Jeśli wybraliśmy wczytanie gry, program prosi o podanie nazwy pliku z starym zapisem gry. Poprawne dane: wszystko. Program nie sprawdza czy nazwa zakończona jest „.txt” lub innym rozszerzeniem. W przypadku nieotwarcia pliku program kontynuuje poprawne działanie. Oznacz to, że program nie kontroluje wprowadzonych danych.
   2. 3.2. Jeśli nie wybraliśmy wczytania gry, program kontynuuje poprawne działanie.
4. Następnie program prosi pierwszego gracza o podanie jego nicku. Poprawne dane: wszystko. Program nie kontroluje wprowadzonych danych.
5. Następnie program prosi o podanie znaku dla pierwszego gracza. Poprawne dane: X lub O. (Wielkość liter ma znaczenie). W przypadku podania błędnych danych, program wyświetla komunikat o błędzie i prosi o ponowne wprowadzenie poprawnych danych. Oznacz to, że program jest przygotowany na błędne dane.
6. Następnie program prosi drugiego gracza podanie nicku. Poprawne dane: wszystko. Program nie kontroluje wprowadzonych danych. (W przypadku gdy w p.1 wybraliśmy grę z komputerem ten punkt jest pomijany. Komputer jest drugim graczem).
7. Następnie gracz wybiera pole gdzie stawia swój znak. Robi to w dwóch etapach. Najpierw podaje współrzędną ‘y’. Poprawne dane: liczby 1, 2 ,3 ,…,15. Potem podaje współrzędną ‘x’. Poprawne dane: literki a, b, c ,…, o. W przypadku podania błędnych danych wyświetla komunikat o błędzie i prosi o ponowne wprowadzenie poprawnych danych. Oznacza to, że program jest przygotowany na błędne dane. (Ten punkt jest powtarzany aż do zakończenia gry).
8. W przypadku wybrania w p.1 opcji grania na dwóch graczy, komputer pokarze nam dodatkową możliwość gry. Możemy zagrać w tryb gry swap. Poprawne dane to: 1 lub 2. W przypadku wpisania jakiejkolwiek litery wyświetla komunikat o błędzie i prosi o ponowne wprowadzenie poprawnych danych. W przypadku wpisania innej cyfry powtarza całą procedurę czyli wcześniejszą prośbę o wybór. Oznacza to, że program jest przygotowany na błędne dane. (W przypadku wybrania tego trybu komputer będzie wyświetlał komunikaty dotyczące wyboru miejsca stawiania znaku dla odpowiedniego gracza, podobnie jak w p. 7).

## Nietypowe zachowania programu

-Nie jestem pewny czy odmiana gry swap jest w pełni zgodna z jej oryginalnymi założeniami. Mimo wszystko starałem się żeby działała logiczne (według mnie), bez oczywistego faworyzowania któregoś z graczy.

-W przypadku gry na dwóch graczy program nie rozpoznaje remisu. Jest tak ze względu na komplikacje związane z trybem swap.

## Niezgodności z założeniami przekazanymi w treści zadania

1. Błędy, braki:

-Przy wyświetlaniu planszy, program na końcu wyświetla ‘32’. Jest to tylko wizualną wadą, nie przeszkadza w rozgrywce ani w działaniu programu.

-Plansza zapisana do pliku „.txt” będzie wyświetlana w nim w jednym wierszu. Nie przeszkadza to w późniejszym wczytaniu gry.

2. Dodatkowe, niewymagane czynności programu:

-Wybranie nazw graczy oraz znaku którym się będzie grało.

-Odmiana gry swap (dla dwóch graczy).

-Zapis oraz wczytanie (wcześniej zapisanego) stanu gry.

- Możliwość grania z komputerem.

# Składnia danych wejściowych i wyjściowych

## Struktura plików, typy danych przechowywane w poszczególnych polach pliku

Pliki „.txt” są zapisywane do podwójnej tablicy znaków o nazwie plansza lub pobierane z podwójnej tablicy znaków o nazwie plansza. Plik (o rozmiarze 18x18 miejsc znakowych) do którego zapisujemy planszę jest tworzony, gdy użytkownik wpisze jego nazwę (z rozszerzeniem .txt). Wtedy do tego pliku zostanie zapisana pusta plansza, taka jak wyświetlona pierwszy raz przez program. Wraz z trwaniem rozgrywki na miejscach znaków „+” będą pojawiały się „X” lub „O”, odpowiednio tam gdzie postawili je gracze. Wszystkie osiemnaście wierszy wyświetlane w programie zostaną zapisane w pliku w jednym wierszu. Będą oddzielone tylko spacjami.

Jeśli użytkownik zdecyduje się wczytać grę będzie musiał podać nazwę istniejącego już pliku (wcześniej utworzonego przez program). Wtedy jest pewność poprawnego wczytania wszystkich znaków z pliku, zapisania ich do zmiennej plansza i kontynuacji gry.

## Struktura katalogów niezbędna dla poprawnego działania programu

Utworzone pliki z zapisanym stanem gry muszą znajdować się w folderze „\gomoku\gomoku” aby działać.

Przykładowym plikiem gotowym do wczytania, znajdującym się już w tym folderze jest plik „save10.txt”.

# Opis kodu

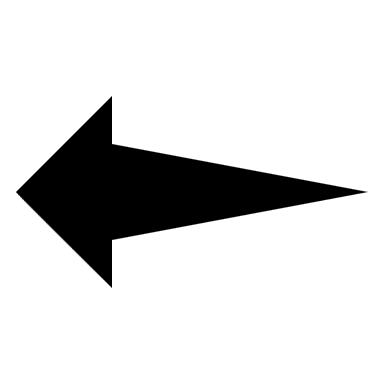
## Lista plików z kodem źródłowym wchodzących w skład programu

gomoku.cpp – zawiera jedną funkcje, konstruktor i main.

gracz\_vs\_gracz.cpp – zawiera definicje metoda z klas gracz\_vs\_gracz, gracz\_vs\_pc oraz save.

gracz\_vs\_gracz.h – zawiera deklaracje klas gracz\_vs\_gracz, gracz\_vs\_pc, save oraz szablon.

## Diagram klas

Klasa główna: class gracz\_vs\_gracz  Przyjaciel klasy gracz\_vs\_gracz:

#gracz1[ ] : char class save

#gracz2[ ] : char #\*stream1 : FILE

#znak1 : char #\*stream2 : FILE

#znak2 : char #nazwa\_pl1[ ] : char

#wygrana : int = 0 #nazwa\_pl2[ ] : char

+wybor\_nick() : int +czy\_wczytac() : int

+plansza[ ][ ] : char +tworz\_plik() : void

#akt\_ruch[ ] : char +zapis(gracz\_vs\_gracz \*g) : void

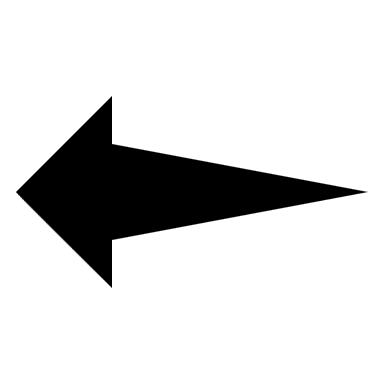
+pokarz() : void +wczytaj(gracz\_vs\_gracz \*g) : int

+ruch(int i) : void

+sprawdzenie() : int

+czy\_swap() : int

+swap() : int



Podklasa: class gracz\_vs\_pc

+wybor\_nick() : int

+ruch(int i) : void