

# ZAAWANSOWANE PROGRAMOWANIE KOMPUTEROWE WNE (2023)

KRZYSZTOF ZIEMIAŃSKI

## 3. PROJEKT ZALICZENIOWY

Praca zaliczeniowa powinna składać się z:

- (1) Programu napisanego w C++ realizującego założone zadanie. Programy powinny być odpowiednio skomentowane: każda funkcja/klasa powinna być opatrzona opisem.
- (2) Dokumentacji do programu. Dokumentacja powinna zawierać:
  - opis zadania realizowanego przez program
  - opis funkcji programu
  - instrukcję użytkownika programu
  - krótki opis kodu programu.
- (3) Opisu eksperymentów (opcjonalnie). Jeśli tematem programu jest przeprowadzenie pewnych symulacji, należy opisać je i wyciągnąć z nich wnioski.

Praca powinna być wykonana przez dwu- lub trzyosobowe zespoły (ewentualnie samodzielnie). Temat pracy można wybrać; idealnie będzie, jeśli napisany program będzie mógł być użyty nie tylko do zaliczenia tego przedmiotu. Przed rozpoczęciem pracy należy zgłosić jej temat do zaakceptowania. Gotowe prace należy wysłać na adres [ziemians@mimuw.edu.pl](mailto:ziemians@mimuw.edu.pl).

Poniżej propozycje tematów na program zaliczeniowy

**Kalkulator.** Napisać program, który umożliwi wyliczanie wartości wyrażeń arytmetycznych wpisywanych do terminala. Należy zadbać o właściwą kolejność działań. Możliwe rozszerzenia: definiowanie własnych zmiennych i funkcji.

**Automaty komórkowe.** Stworzyć symulator gry z życia [https://en.wikipedia.org/wiki/Conway's\\_Game\\_of\\_Life](https://en.wikipedia.org/wiki/Conway's_Game_of_Life). Plansza podzielona jest na kwadratowe pola. W niektórych polach znajdują się komórki. Jeśli komórka ma mniej niż dwóch lub więcej niż trzech sąsiadów, znika; jeśli wolne pole ma dokładnie trzech sąsiadów, pojawia się na nim nowa komórka. Program powinien umożliwiać przeprowadzenie symulacji dla różnych początkowych układów. Możliwe warianty: inne gry komórkowe, np. mrówki Langtona.

**Układanka.** Dany jest pewien wzorzec składający się z przylegających do siebie kwadratów o jednakowych rozmiarach, oraz pewien zestaw klocków (również składających się z kwadratów). Należy stworzyć program, który znajduje pokrycia wzorca (jedno lub wszystkie) przy pomocy nienachodzących na siebie klocków. Możliwe warianty: rozpatrywać siatki trójkątne lub sześciokątne. Więcej szczegółów: np. <http://en.wikipedia.org/wiki/Pentomino>.

**Dylemat więźnia.** Pewna ilość graczy ( $n$ ) w każdej rundzie gra w dylemat więźnia, każdy z każdym pewną ilość rund ( $l$ ). Po każdej rundzie kilku graczy ( $k$ ), którzy uzyskali najgorsze wyniki umiera, a ich miejsce zajmuje potomstwo  $k$  najlepszych graczy. Każdy z graczy stosuje ustaloną strategię - dokonuje wyboru z prawdopodobieństwem określonym tylko i wyłącznie na podstawie ostatnich  $r$  ruchów przeciwnika. Potomek zwycięskiego gracza otrzymuje nieznacznie zaburzoną strategię rodzica. Należy napisać program, który wykonuje symulacje i umożliwi dokonanie ich analizy (tj. stwierdzenie, jakie strategie odniosły sukces). Można też rozważać inne gry, np. papier-kamień-nożyce.

**Korelacje.** Należy stworzyć program, który wyszukuje korelacje pomiędzy zmianami kursów akcji w poszczególnych dniach. Przetestować program i wyciągnąć wnioski przy użyciu rzeczywistych danych. Możliwe warianty: przetestować w podobny sposób inne dane, np. meteorologiczne.

**Kompresja tekstów.** Zaimplementować program wykonujący kompresję i dekompresję tekstów metodą Huffmana (zob. [http://en.wikipedia.org/wiki/Huffman\\_coding](http://en.wikipedia.org/wiki/Huffman_coding)). Opracować optymalne kody do kodowania tekstów w różnych językach. Sprawdzić, czy zapisywanie niektórych często występujących słów jako pojedynczych symboli zwiększa efektywność kodowania.

**Symulacja rozkładu bogactwa.** Pewna ilość osób na początku symulacji posiada jednaki majątek. W każdej rundzie symulacji wybierane są losowo dwie osoby, które zawierają transakcję. W efekcie transakcji jedna z tych osób zyskuje od drugiej pewną kwotę (osoba wygrywająca jest losowana). Wielkość tej kwoty to pewna część (np. 1%) majątku uboższej osoby. Sprawdzić, jak kształtuje się rozkład bogactwa po wykonaniu pewnej liczby rund. Następnie przetestować w jaki sposób na rozkład bogactwa wpływają np. zdolności poszczególnych osób (osoby zdolniejsze częściej "wygrywają" transakcje) lub programy socjalne (np. po pewnej ilości rund wszyscy uczestnicy płacą 10% podatku, który jest rozdawany po równo wszystkim uczestnikom). Więcej: <https://www.scientificamerican.com/article/is-inequality-inevitable>.

**Grafy.** Graf składa się ze zbioru wierzchołków i krawędzi, które łączą pewne pary wierzchołków. Graf jest spójny, jeśli każde dwa wierzchołki można połączyć pewną sekwencją krawędzi. Dwa grafy  $G$  i  $H$  są izomorficzne jeśli można znaleźć bijekcję (funkcję różnowartościową i na)  $f : G \rightarrow H$  o tej własności, że  $g_1, g_2 \in G$  są połączone wtedy i tylko wtedy, gdy  $f(g_1)$  i  $f(g_2)$  są połączone. Napisać program, który wylicza liczbę różnych (tj. nieizomorficznych) spójnych grafów o zadanej liczbie wierzchołków. Możliwe warianty: ten sam problem dla grafów skierowanych.