Tematy projektów na DPPR

Uwaga! Każdy program który powstanie jako realizacja któregokolwiek projektu powinien być "idiotoodporny" i wykorzystywać wszystkie poznane techniki programistyczne. Ponadto powinien być zgodny ze "sztuką", a więc np. nie mieszać obsługi I/O przez strumienie z tą opartą na scanf/printf. Co więcej nie należy tworzyć statycznych tablic o zmiennym rozmiarze. Aplikacje powinny mieć odpowiednio dobrane, czytelne i funkcjonalne interfejsy (np. wyświetlone tabelki wyników, lub "graficzne" reprezentacje niektórych obiektów). Wymagania dodatkowe nie są opcjonalne, lecz obowiązkowe.

Projekt 1 (Gra w kości)**.** *Napisz program, który umożliwiać będzie grę w kościanego pokera. Program powinien umożliwiać grę pojedynczą i przeciwko innemu graczowi.*
Dodatkowe wymagania:

- *Pomoc użytkownika w postaci skróconych zasad. Pomoc ta powinna być podmienialna bez konieczności ponownej kompilacji programu.*
- *Lista najlepszych wyników trzymana w zewnętrznym pliku.*
- *Graficzne przedstawienie tabelki z aktualnymi wynikami z "podświetlonymi" możliwymi wynikami aktualnego rzutu. (Graficzne przedstawienie kości mile widziane.)*

Projekt 2 (Gra w wojnę)**.** *Napisz program, który umożliwiać będzie grę w wojnę. Wymagane są co najmniej dwie funkcjonalności: gra z komputerem oraz automatyczne rozstrzygnięcie rozgrywki dla podanych/losowych permutacji talii kart. Gracz w każdej chwili musi wiedzieć ile kart zostało jeszcze w jego talii, a ile w talii przeciwnika.*
Dodatkowe wymagania:

- *Pomoc użytkownika w postaci skróconych zasad. Pomoc ta powinna być podmienialna bez konieczności ponownej kompilacji programu.*
- *Trzymane w zewnętrznym pliku wyniki automatycznie rozegranych rozgrywek umożliwiające ustalenie zwycięzcy bez konieczności ponownego rozgrywania. całej rozgrywki.*

- *Graficzne przedstawienie kart, które są na stole.*

Projekt 3 (Gra w statki)**.** *Napisz program, który umożliwiać będzie grę w statki. Wymagana jest możliwość rozgrywki z grającym losowo* komputerem. Gwiazdka przy "losowo" oznacza, że komputer owszem gra losowo, ale nie głupio, tj. jeśli trafi w pole z fragmentem statku, to będzie się starał go zatopić strzelając na około, a nie w zupełnie inne miejsce.*
Dodatkowe wymagania:

- *Pomoc użytkownika w postaci skróconych zasad. Pomoc ta powinna być podmienialna bez konieczności ponownej kompilacji programu.*
- *Lista najlepszych wyników trzymana w zewnętrznym pliku, gdzie przez lepsze wyniki rozumiemy te, w których zwycięstwo nastąpiło po mniejszej ilości ruchów.*
- *Graficzne przedstawienie planszy gracza i przeciwnika.*

Projekt 4 (Blackjack)**.** *Napisz program, który umożliwia grę w Blackjacka (zasady można znaleźć na http://pl.wikipedia.org/wiki/Blackjack) z komputerowym krupierem. Gracz powinien na początku tworzyć swój profil (zapisywany do pliku i zawierający dotychczasowy wynik) otrzymując pewną bazową ilość wirtualnych pieniędzy. Gracz powinien mieć również możliwość zapisu stanu gry (tzn. swojego stanu posiadania po zakończonym rozdaniu) i odtworzenia jej później.*
Dodatkowe wymagania:

- *Pomoc użytkownika w postaci skróconych zasad. Pomoc ta powinna być podmienialna bez konieczności ponownej kompilacji programu.*
- *Graficzne przedstawienie kart (uproszczony sposób) na ekranie.*

Projekt 5 (Poker)**.** *Napisz program umożliwiający grę w Pokera z wirtualnym przeciwnikiem. Tak jak w projekcie Blackjack gracz powinien najpierw tworzyć profil i otrzymywać pewną pulę wirtualnych pieniędzy. Następnie gracz może rozpocząć rozgrywki. Każde rozdanie powinno przebiegać wg. następującego schematu:*

1. *Gracze wkładają do puli pewną ustaloną kwotę pieniędzy.*
2. *Obaj gracze otrzymują po 5 kart.*
3. *Gracze decydują, czy chcą dołożyć jakąś kwotę do puli (dla uproszczenia można ustalić możliwą kwotę pojedynczego podniesienia).*
4. *Gracze tak długo dokładają do puli, aż któryś z nich nie spasuje, albo obaj nie ustalą jednakowej wartości podniesienia.*
5. *Gracze wybierają karty, które chcą wymienić.*
6. *Gracze powtarzają kroki 3 i 4.*
7. *Gracz o niższym wyniku, lub ten, który spasował traci całą kwotę włożoną do puli, a jego przeciwnik wzbogaca się o nią.*

"Sztuczną inteligencję" komputera można zawrzeć w kilku sensownych regułach, których będzie się trzymał (do uzgodnienia z prowadzącym).
Dodatkowe wymagania:

- *Pomoc użytkownika w postaci skróconych zasad. Pomoc ta powinna być podmienialna bez konieczności ponownej kompilacji programu.*
- *Graficzne przedstawienie kart (uproszczony sposób) na ekranie.*

Projekt 6 (Kalkulator)**.** *Napisz program, który umożliwi obliczanie wartości wyrażeń arytmetycznych wpisywanych z klawiatury. Obowiązkowo obsługiwać musi następujące działania: dodawanie, odejmowanie, mnożenie, dzielenie, potęgowanie (dowolne podstawy i wykładniki), funkcje trygonometryczne (z argumentami podawanymi w stopniach). Formuły mogą być złożone i zawierać nawiasy (np. 3+(4$\hat{7}$)-pi/(e$\hat{9}$)+(cos(30))$\hat{}$(2.04)+sin(13)). Przy przetwarzaniu formuły przydatne może być sprowadzenie jej do ONP (http://pl.wikipedia.org/wiki/Odwrotna_notacja_polska).*
Wymagania dodatkowe:

- *Pomoc użytkownika objaśniająca zasady używania kalkulatora. Pomoc ta powinna być podmienialna bez konieczności ponownej kompilacji programu.*

- *Trzymana w pliku zewnętrznym "pamięć" operacji wykonanych od czasu uruchomienia programu. Do pamięci tej użytkownik powinien móc się odwoływać (przeglądać ją i ponownie wykorzystywać).*

Projekt 7 (Sklep)**.** *Napisz program umożliwiający obsługę sklepu. Program powinien umożliwiać:*

1. *Dodanie/usunięcie/modyfikację towaru (opisywanego przez numer, nazwę i producenta) do bazy towarów.*
2. *Dodanie towaru pochodzącego z bazy towarów do magazynu (od tego momentu towar opisywany jest dodatkowo przez cenę nabycia, cenę w sklepie, datę nabycia, datę sprzedaży).*
3. *Usunięcie/modyfikację towaru w magazynie.*
4. *Sprzedaż towaru.*
5. *Zakup towaru z hurtowni.*
6. *Przeglądanie danych księgowych, tj. rozchodów i przychodów.*
7. *Przeglądanie danych dot. towarów pod kątem różnych ich parametrów (np. wszystkich towarów zakupionych od 10.10.2011 do 11.11.2011).*
8. *Inne uzgodnione z prowadzącym.*

Dodatkowe wymagania:

- *Pomoc użytkownika objaśniająca zasady używania programu. Pomoc ta powinna być podmienialna bez konieczności ponownej kompilacji programu.*
- *Wszystkie "bazy danych" winny znajdować się w plikach zewnętrznych.*
- *Dane wyświetlane powinny mieć sensowną strukturę (np. odpowiednich tabel).*

Projekt 8 (Biblioteka)**.** *Napisz program służący do obsługi biblioteki. Program powinien umożliwiać:*

1. *Zarządzanie zbiorem książek - dodawanie/usuwanie/modyfikowanie (książka powinna być definiowana co najmniej przez tytuł, nr. ISBN, autora, wydawnictwo, rok wydania).*

2. *Zarządzanie użytkownikami - dodawanie/usuwanie/modyfikacja (użytkownik powinien być definiowany co najmniej przez imię, nazwisko, nr PESEL)*

3. *Zarządzanie wypożyczeniami i zwrotami książek (terminy wypożyczeń i zwrotów itp.)*

4. *Przeglądanie danych archiwalnych.*

5. *Inne uzgodnione z prowadzącym.*

Dodatkowe wymagania:

- *Pomoc użytkownika objaśniająca zasady używania programu. Pomoc ta powinna być podmienialna bez konieczności ponownej kompilacji programu.*

- *Wszystkie "bazy danych" winny znajdować się w plikach zewnętrznych.*

- *Dane wyświetlane powinny mieć sensowną strukturę (np. odpowiednich tabel).*

Projekt 9 (ZOO)**.** *Napisz program pomagający w zarządzaniu ZOO. Program powinien umożliwiać zarządzanie zasobami zwierzęcymi oraz lokalowymi, w tym także przydział zwierząt (definiowanych przez gatunek, płeć, imię, numer, minimalną potrzebną przestrzeń, akceptowalne typy klatek itp) do klatek i wybiegów (definiowanych przez rozmiar i typ). Pozostałe wymagania analogicznie, jak w programie Biblioteka.*

Projekt 10 (Zawody)**.** *Napisz program pomagający w organizowaniu zawodów szachowych. Program musi umożliwiać zarządzanie zawodami (określanymi przez nazwę, ilość graczy, ilość rund, ranking graczy), graczami (definiowanymi przez imię, nazwisko, wiek, numer, ilość zwycięstw, ilość remisów i ilość porażek), a także automatyczne parowanie graczy na kolejne rundy, by*

najlepsi grali z najlepszymi (ale bez powtórzeń). Po zakończonych zawodach powinny być one archiwizowane w pliku zewnętrznym. Program powinien również umożliwiać przeglądanie danych archiwalnych.
Dodatkowe wymagania:

- *Pomoc użytkownika objaśniająca zasady używania programu. Pomoc ta powinna być podmienialna bez konieczności ponownej kompilacji programu.*
- *Dane wyświetlane powinny mieć sensowną strukturę (np. odpowiednich tabel).*

Projekt 11 (Organizer)**.** *Napisz program pomagający w organizacji codziennego życia. Program powinien umożliwiać przeglądanie kalendarza, dodawanie wydarzeń/przypomnień do niego, a także sprawdzanie, czy od zeszłego uruchomienia programu któreś ze zdarzeń miało miejsce. Zdarzenia powinny być charakteryzowane przynajmniej przez nazwę, numer, datę rozpoczęcia/zakończenia, godzinę rozpoczęcia/zakończenia i komentarz.*
Dodatkowe wymagania:

- *Pomoc użytkownika objaśniająca zasady używania programu. Pomoc ta powinna być podmienialna bez konieczności ponownej kompilacji programu.*
- *Zdarzenia w kalendarzu powinny być zapamiętywane w zewnętrznym pliku.*
- *Dane wyświetlane powinny mieć sensowną strukturę (np. odpowiednich tabel symulujących kalendarz).*

Projekt 12 (Gra Othello)**.** *Napisz program pozwalający rozgrywać pojedynki w grze Othello. Wymagana jest co najmniej możliwość gry przeciwko innemu graczowi i komputerowi stosującemu jakąś prymitywną strategię (do uzgodnienia z prowadzącym). Więcej informacji o grze można uzyskać na http://othellomania.pl/. Program powinien umożliwiać tworzenie profili graczy (zawierających poza nazwą gracza także statystyki jego rozgrywek), które trzymane byłyby w zewnętrznych plikach.*
Dodatkowe wymagania:

- *Pomoc użytkownika w postaci skróconych zasad. Pomoc ta powinna być podmienialna bez konieczności ponownej kompilacji programu.*
- *Graficzne przedstawienie planszy z uwzględnieniem różnych kolorów (uproszczony sposób) na ekranie.*

Projekt 13 (Interaktywna książka kucharska)**.** *Napisz program, który symulować będzie książkę kucharska z wyszukiwarką przepisów. Program powinien umożliwiać zarządzanie przepisami (trzymanymi w zewnętrznych plikach), a także wyszukiwanie przykładowych przepisów na podstawie kilku podanych składników.*
Dodatkowe wymagania:

- *Pomoc użytkownika objaśniająca zasady używania programu. Pomoc ta powinna być podmienialna bez konieczności ponownej kompilacji programu.*
- *Dane wyświetlane powinny mieć sensowną strukturę (np. odpowiednich tabel).*

Projekt 14 (Sudoku)**.** *Napisz program umożliwiający grę w Sudoku na losowo wygenerowanych planszach oraz na tych zapisanych w zewnętrznych plikach. Program powinien uniemożliwiać edycję początkowo wypełnionych pól oraz wyświetlać planszę w sposób pozwalający odróżnić je od pól wypełnianych przez gracza.*
Dodatkowe wymagania:

- *Pomoc użytkownika objaśniająca zasady gry. Pomoc ta powinna być podmienialna bez konieczności ponownej kompilacji programu.*

Projekt 15 (Grafy)**.** *Napisz program pozwalający reprezentować grafy w pamięci komputera za pomocą macierzy sąsiedztwa oraz wykonywanie na nim algorytmów BFS i DFS. Program powinien umożliwiać zapis grafu w pliku zewnętrznym i wyświetlanie jego macierzy.*
Dodatkowe wymagania:

- *Pomoc użytkownika objaśniająca zasady obsługi programu i opisy algorytmów BFS i DFS. Pomoc ta powinna być podmienialna bez konieczności ponownej kompilacji programu.*

Projekt 16 (Gra w kółko i krzyżyk 3D)**.** *Napisz program, który umożliwiać będzie grę w kółko i krzyżyk na trójwymiarowych planszach 3x3x3 i 4x4x4. Wymagana jest możliwość rozgrywki z grającym losowo/prostą strategią komputerem lub drugim graczem.*
Dodatkowe wymagania:

- *Pomoc użytkownika w postaci skróconych zasad. Pomoc ta powinna być podmienialna bez konieczności ponownej kompilacji programu.*
- *Ranking rozgrywek pomiędzy parami graczami trzymany w zewnętrznym pliku. Rankingi powinny być uaktualniane w przypadku gdy dwóch graczy gra ze sobą ponownie.*
- *Graficzne przedstawienie trzech/czterech poziomów planszy.*

Projekt 17 (Kostka Rubika)**.** *Napisz program, który będzie umożliwiać układanie kostki Rubika 3x3x3. Wymagane jest generowanie losowego ułożenia początkowego kostki.*
Dodatkowe wymagania:

- *Pomoc użytkownika w postaci skróconych zasad. Pomoc ta powinna być podmienialna bez konieczności ponownej kompilacji programu.*
- *Lista najlepszych wyników trzymana w zewnętrznym pliku, gdzie przez lepsze wyniki rozumiemy te, w których kostka została ułożona po mniejszej ilości ruchów.*
- *Graficzne przedstawienie kostki.*

Projekt 18 (Szyfrator/deszyfrator)**.** *Napisz program, który umożliwiać będzie szyfrowanie i deszyfrowanie przy pomocy algorytmów: AES, 3DES, IDEA i Twofish. Wymagane jest aby teksty jawne były wczytywane z pliku tekstowego a szyfrogramy zapisywane do pliku binarnego.*
Dodatkowe wymagania:

- *Pomoc użytkownika w postaci opisu działania programu. Pomoc ta powinna być podmienialna bez konieczności ponownej kompilacji programu.*

- *Szyfrator/deszyfrator ma umożliwiać działanie w trybach: ECB, CBC i CFB dla wiadomości/szyfrogramów o długości większej niż wielkość bloku.*

- *Informacje o użytym algorytmie i trybie powinny być zapisane razem z szyfrogramem w jednym pliku.*

Projekt 19 (Generator haszy)**.** *Napisz program, który umożliwiać będzie generowanie i weryfikację haszy zadanych wiadomości przy pomocy finałowych kryptograficznych funkcji haszujących w konkursie SHA-3. Program powinien wczytywać wiadomości z wybranego pliku oraz opcjonalnie umożliwiać podanie klucza użytkownika i generowanie tzw. Hash-based Message Authentication Code.*
Dodatkowe wymagania:

- *Pomoc użytkownika w postaci opisu działania programu. Pomoc ta powinna być podmienialna bez konieczności ponownej kompilacji programu.*

- *Hash-based Message Authentication Code powinien być generowany według np. standardu RFC 2104.*

- *Hasz oraz informacje o wykorzystanym algorytmie powinny być zapisywane w jednym pliku.*

Projekt 20 (Łamacz szyfrów)**.** *Napisz program, który umożliwiać będzie szyfrowanie wiadomości i deszyfrowanie szyfrogramów przy pomocy szyfrów Cezara i Vigenere'a. Program oprócz analogicznych funkcjonalności opisanych w P3 powinien umożliwiać łamanie pojedynczych szyfrogramów bez danego klucza szyfrowania.*
Wymagania dodatkowe:

- *Pomoc użytkownika w postaci opisu działania programu. Pomoc ta powinna być podmienialna bez konieczności ponownej kompilacji programu.*

- *Informacje o użytym algorytmie powinny być zapisane razem z szyfrogramem w jednym pliku.*

- *Wskazane jest zastosowanie metod statystycznych kryptoanalizy szyfrów podstawieniowych.*

Projekt 21 (Aplikacja do wystawiania/rezerwowania miejsc hotelowych)**.** *Napisz program umożliwiający z jednej strony właścicielowi hotelu/pensjonatu wystawienie oferty, a z drugiej użytkownikowi rezerwacje oferty.*
Wymagane funkcje:

1. *Uproszczone tworzenie kont użytkowników dwóch rodzajów: oferującego i rezerwującego (użytkownicy powinni być definiowani przez nazwę, hasło oraz typ konta).*
2. *Uproszczony system logowania.*
3. *Wystawianie ofert (definiowanych przez: nazwę hotelu, ilość miejsc, cene za miejsce, początku i końca oferty i stan - wolna, zarezerwowana, sprzedana) przez właściciela.*
4. *Rezerwowanie ofert przez użytkownika.*
5. *Odrzucenie przez oferującego rezerwacji.*
6. *Wyświetlanie danych dotyczących ofert z możliwością wyboru rodzaju wyświetlanych ofert.*
7. *Inne uzgodnione z prowadzącym.*

Dodatkowe wymagania:

- *Pomoc użytkownika objaśniająca zasady używania programu. Pomoc ta powinna być podmienialna bez konieczności ponownej kompilacji programu.*
- *Wszystkie "bazy danych" winny znajdować się w plikach zewnętrznych.*
- *Każde konto użytkownika powinno być zapisywane do pliku.*
- *Dane wyświetlane powinny mieć sensowną strukturę (np. odpowiednich tabel).*

Projekt 22 (Elektroniczny dziekanat)**.** *Napisz program "ułatwiający" pracę dziekanatowi. Program umożliwiać powinien:*

1. *Zarządzanie zasobami ludzkimi: studentami (definiowanymi przez imię, nazwisko, PESEL, rok studiów), pracownikami naukowo-dydaktycznymi (definiowanymi przez imię, nazwisko, PESEL, zakład, stanowisko i stopień/tytuł naukowy).*

2. *Zarządzanie przedmiotami (definiowanymi przez nazwę i ilość punktów ECTS).*

3. *Zarządzanie grupami przedmiotowymi (definiowanymi przez identyfikator i przedmiot).*

4. *Dodawanie i usuwanie prowadzących/studentów do grup.*

5. *Wyświetlanie list studentów na danym przedmiocie w danej grupie, przedmiotów, na które chodzi dany student, przedmiotów prowadzonych przez danego prowadzącego itp.*

6. *Inne uzgodnione z prowadzącym.*

Dodatkowe wymagania:

- *Pomoc użytkownika objaśniającą zasady używania programu.*
- *Pomoc ta powinna być podmienialna bez konieczności ponownej kompilacji programu.*
- *Wszystkie "bazy danych" winny znajdować się w plikach zewnętrznych.*

Projekt 23 (Bank)**.** *Napisz program umożliwiający obsługę banku. Program powinien pozwalać na:*

- *Zarządzanie kontami klientów indywidualnych (definiowanymi przez nazwisko, imię, PESEL, nr konta i oprocentowanie).*
- *Zarządzanie kontami klientów nie-indywidualnych (definiowanymi przez nazwę, NIP, nr konta i oprocentowanie).*

- *Przydzielanie kredytów (definiowanych przez nazwę, kwotę początkowa, kwotę pozostała do spłacenia, wysokość raty i oprocentowanie) klientom. Oraz zarządzanie nimi.*
- *Obliczanie ilości środków na koncie po ustalonym okresie czasu (z uwzględnieniem oprocentowania konta i ewentualnym, terminowym spłacaniu rat kredytów).*
- *Wyświetlanie danych statystycznych dot. klientów, kont, kredytów itp.*
- *Inne uzgodnione z prowadzącym.*

Dodatkowe wymagania:

- *Pomoc użytkownika objaśniająca zasady używania programu. Pomoc ta powinna być podmienialna bez konieczności ponownej kompilacji programu.*
- *Wszystkie "bazy danych" winny znajdować się w plikach zewnętrznych.*
- *Dane wyświetlane powinny mieć sensowna strukturę (np. odpowiednich tabel).*

Projekt 24 (Gra typu MUD)**.** *Napisz prosta tekstowa grę przygodowa typu MUD. Gra powinna pozwalać na proste podmienianie lokacji, dodawanie przeciwników, przedmiotów itp, bez konieczności kompilowania programu, oraz prowadzenie turowej rozgrywki w wygenerowanym świecie. Z racji specyfikacji projektu, konkretne wymagania dostępne u prowadzącego.*

Projekt 25 (Kalkulator pochodnych)**.** *Napisz program umożliwiający obliczanie pochodnych podanych funkcji. Wymaganymi funkcjami są: wielomiany, funkcje trygonometryczne, logarytm naturalny, funkcje wykładnicze oraz ich sumy, różnice, iloczyny, ilorazy (odwrotności dla chętnych). W programie można założyć, ze podane funkcje będą miały odpowiednio dobra strukturę, tj. tyle nawiasów ile potrzeba w celu łatwego przetwarzania funkcji. Jednakże musi istnieć możliwość przywołania ostatnio wpisanej formuły, nawet jeśli była błędna. Program powinien umożliwiać również wczytywanie funkcji z plików tekstowych. Pozostałe wymagania analogiczne do programu Kalkulator (poza pamięcią w pliku).*