Techniki programowania – projekt, Informatyka, sem. 2, studia niestacjonarne

- W ramach projektu należy zrealizować zadanie poprzez napisanie, uruchomienie, przetestowanie i przesłanie prowadzącemu do sprawdzenia programu w języku C lub C++ o wybranej tematyce, wraz z odpowiednią przygotowaną dokumentacją.
- Program może być napisany obiektowo bądź nie obiektowo, można używać dowolnych elementów języka programowania C/C++. Nowszych elementów języka, wprowadzonych np. do specyfikacji C++ 2011 czy 2014 czy 2017, można używać, ale oczywiście nie trzeba. Przy czym, generalnie należy rozumieć kod, który się napisało. Lepiej zrealizować coś prościej, ale tak, że się daną konstrukcję/składnię rozumie, a nie tak, że się fragment kodu automatycznie skądś skopiowało/kolega napisał itp. Prowadzący zastrzega sobie prawo do poproszenia o dokładne wyjaśnienie działania wskazanych fragmentów kodu.
- Program generalnie powinien być prosty, jednak na tyle skomplikowany, aby zasadne było zawarcie jego kodu źródłowego w co najmniej 2 modułach, czyli jednostkach kompilacji – tj. program główny i co najmniej jeden jeszcze moduł; każdy moduł, oprócz programu głównego, powinien składać się z 2 plików: *.h i *.c / *.cpp o tej samej nazwie.
- Zaleca się, żeby program był aplikacją konsolową i jak najmniej wchodził w interakcję z użytkownikiem; zaleca się, w miarę możliwości, żeby wszystkie dane wejściowe były wprowadzane do programu z linii komend jako jego parametry.
- Można samodzielnie zaproponować temat zadania bądź wybrać program z listy przedstawionej przez prowadzącego (w drugim załączonym pliku).
- 2 osoby mogą razem realizować 1 zadanie.
- Użyte środowisko programistyczne oraz system operacyjny, pod którym ono działa – dowolne. Należy zadbać o to, żeby prowadzący mógł łatwo obejrzeć zarówno kod źródłowy jaki i program w działaniu. Nie musi kod kompilować się przy prowadzącym. Nie należy zakładać, że prowadzący zapewni u siebie na komputerze to środowisko, w którym pisaliśmy program; prowadzący używa Dev-C++ lub CodeBlocks pod Windows.
- Rozliczanie się z zadania składa się z następujących etapów:

  1) Ustalenie (wybór z załączonej listy bądź zaproponowanie własnego) zadania wraz z jego zwięzłą, kilkuzdaniową specyfikacją (m. in. określenie co program robi, określenie formatu danych wejściowych itp.).

  2) Podział zadania na moduły (jednostki kompilacji) z opisem każdego modułu, diagram hierarchii modułów (tj. schemat pokazujący jak poszczególne moduły zależą podczas kompilacji od siebie), pseudokod programu oraz schematy blokowe pokazujące główną logikę programu.

  3) Wstępna wersja działająca programu.

  4) Końcowa wersja działającego programu. Plik README opisujący program. Opcjonalnie: plik makefile.

- Terminy realizacji kolejnych etapów projektu ustalane są przez prowadzącego na początku semestru. Wszystkie etapy trzeba zaliczyć, natomiast ocena wystawiana jest na zakończenie za całość projektu. Możliwe jest wcześniejsze rozliczenie się z danego etapu bądź z całości projektu.
- Kryteria brane pod uwagę przy ocenie programu są następujące: czy program działa poprawnie, czy w sposób zrozumiały komunikuje się z użytkownikiem (prowadzący musi

w łatwy sposób poradzić sobie samodzielnie z przetestowaniem całości programu), czy jest odpowiednia kontrola poprawności danych wejściowych, czy kod jest przejrzysty i czytelny (np. obecność wcięć, komentarzy itp.), samodzielność realizacji zadania, kompletność przedstawionej dokumentacji, terminowość.

Techniki programowania, projekt

Informatyka, studia niestacjonarne, semestr 2

Przykładowe propozycje zadań

1. Dla zadanych współczynników a_i, b_i, c_i, d_i wyznaczyć rozwiązanie układu trzech równań liniowych:

   Sprawdzić, czy układ jest oznaczony, nieoznaczony czy sprzeczny. Generować wyniki liczbowe tylko dla pierwszego z tych przypadków.

2. Dla zadanych współczynników a_i, b_i, c_i, wyznaczyć rozwiązanie układu dwóch równań liniowych:

   Sprawdzić, czy układ jest oznaczony, sprzeczny czy nieoznaczony, a w tym ostatnim przypadku, czy rozwiązanie zależne jest od jednego czy od dwóch parametrów. Rozwiązanie podać w odpowiedni sposób w każdym z przypadków (oprócz układu sprzecznego), np. $x = 2$, $y = 1$ (układ oznaczony); $x = t$, $y = 2t + 1$ (układ nieoznaczony, rozwiązanie zależne od jednego parametru).

3. Dla zadanego przedziału liczb rzeczywistych z podanym krokiem obliczyć z zadaną dokładnością l miejsc po przecinku wartość funkcji $cos(x)$. Nie korzystać z gotowej funkcji obliczającej $cos(x)$. Skorzystać z rozwinięcia funkcji $cos(x)$ w szereg Taylora.

4. Dla zadanego przedziału liczb rzeczywistych z podanym krokiem obliczyć z zadaną dokładnością l miejsc po przecinku wartość funkcji $sin(x)$. Nie korzystać z gotowej funkcji obliczającej $sin(x)$. Skorzystać z rozwinięcia funkcji $sin(x)$ w szereg Taylora.

5. Kalkulator macierzowy – realizacja dodawania, odejmowania i mnożenia macierzy (o określonym rozmiarze, np. 3 x 3) oraz mnożenia macierzy przez liczbę.

6. Kalkulator działający na wektorach w przestrzeni trójwymiarowej – realizacja dodawania i odejmowania wektorów, ich iloczynu skalarnego i wektorowego itp.

7. Kalkulator zwykły (działający na zwykłych liczbach skalarnych, rzeczywistych) – np. 4 podstawowe działania arytmetyczne.

8. Obliczanie wyznacznika i macierzy odwrotnej do zadanej macierzy o rozmiarze 4 x 4.

9. Gra w odgadywanie przez użytkownika kolejnego wyrazu ciągu liczbowego na podstawie podanych kilku wyrazów. Generowane będą tylko ciągi arytmetyczne i geometryczne (ew. też inne, zgodnie z inwencją autora programu). Komputer losuje, czy ma wygenerować ciąg arytmetyczny czy geometryczny, a następnie losuje ich parametry: pierwszy wyraz oraz krok bądź iloraz, i generuje kilka pierwszych wyrazów ciągu. Zadaniem użytkownika jest odgadnąć kolejny wyraz.

10. Odgadywanie przez komputer kolejnego wyrazu ciągu liczbowego na podstawie podanych kilku kolejnych wyrazów przez użytkownika, np. trzech. Komputer próbuje rozpoznać ciąg arytmetyczny bądź geometryczny we wprowadzonych wartościach (ew. też inne, zgodnie z inwencją autora programu) i na tej podstawie obliczyć kolejny wyraz.
11. Gra użytkownika z komputerem o następujących zasadach: Gracz rozpoczynający podaje liczbę z zakresu od 1 do 10. Kolejno gracze naprzemian podają liczby, z których każda musi być większa od poprzedniej co najmniej o 1, a co najwyżej o 10. Wygrywa ten, kto poda liczbę 100. Powinno być możliwe określenie, kto ma rozpoczynać oraz jaki ma być poziom trudności, tj. od którego momentu komputer ma zacząć podawać, w miarę możliwości, tylko wyrazy z następującego ciągu: 1, 12, 23, ..., 78, 89.
12. Zdefiniować w programie mały słownik ok. 20 wyrazów w jakimś języku, np. polskim lub angielskim, i napisać procedurę sprawdzającą poprawność pisowni tekstu (słów) zawartego w łańcuchu. Słowa w łańcuchu powinny być wyodrębniane następująco: Jako znaki rozdzielające kolejne słowa traktować znaki: spacji, tabulacji oraz przestankowe: kropka, przecinek, dwukropek, średnik, myślnik/minus, wykrzyknik, znak zapytania, alef (@ - „małpa”), and (&), nawiasy, plus, łamane (*slash*), cudzysłów lub dowolne złożone z nich sekwencje. Inne znaki traktować jako wchodzące w skład słów. Program powinien przeanalizować łańcuch i sporządzić raport z wykazem słów oraz informacją, czy słowo jest poprawne (znajduje się w słowniku), czy błędne.
13. Zdefiniować w programie mały słownik dla ok. 20 wyrazów, np. angielsko-polski lub polsko angielski, i na tej podstawie dokonywać tłumaczenia kolejnych wyrazów w łańcuchu z jednego języka na drugi. Aby słowo mogło zostać przetłumaczone, musi być identyczne z istniejącym w słowniku. W przypadku braku danego słowa z łańcucha w słowniku, wstawiać w jego miejsce określony tekst, np. „%nieznane%”. Znaki rozdzielające pozostawić w nie zmienionej formie.
14. Wyjustować tekst w łańcuchu. Najpierw wyodrębnić słowa zakładając, że znakami rozdzielającymi je mogą być tylko spacje, a wszystkie inne znaki wchodzą w skład słów. Założyć, że znaków tabulacji nie ma w łańcuchu w ogóle. Dla zadanej w programie liczby stanowiącej szerokość strony w znakach (większej, niż długość całego łańcucha), rozsunąć słowa w łańcuchu wstawiając równomiernie dodatkowe spacje pomiędzy nie, tak aby cały łańcuch przyjął wskazaną długość. W przypadku, gdy niemożliwe jest, aby wszystkie przerwy pomiędzy słowami miały jednakową szerokość (ilość spacji), różnica długości pomiędzy najkrótszą i najdłuższą przerwą nie może być większa niż 1 spacja. Sekwencja krótszych przerw powinna wtedy znajdować się w lewej części łańcucha, a sekwencja dłuższych przerw - w prawej. Jeżeli przed rozpoczęciem przetwarzania łańcucha znajdują się w nim spacje wielokrotne, należy je najpierw zastąpić pojedynczymi.
15. Kompresja danych.
16. Szyfrowanie tekstu.
17. Parser wyrażeń arytmetycznych (jak najprostszy).
18. Obliczanie pól, objętości itp., figur, brył geometrycznych itp. (w jednym programie powinna być możliwość realizacji takich obliczeń dla wielu różnych figur).

Można przedstawić także własną propozycję tematu zadania semestralnego.