Wojskowa Akademia Techniczna

Wydział Elektroniki

Programowanie w języku C/C++ dla zastosowań sieciowych - cz. 2

Zadania laboratoryjne

Opracował:
ppłk dr inż. Jarosław Krygier
mgr inż. Sebastian Szwaczyk
mgr inż. Cezary Wilkowski

Koordynował:
ppłk dr inż. Jarosław Krygier

Warszawa 2017

Spis treści

Przygotowanie do ćwiczeń laboratoryjnych:

1. Pobrać, zainstalować i zapoznać się z oprogramowaniem Eclipse.
2. Zapoznać się z materiałami z wykładów - samodzielnie przećwiczyć zadania dodatkowe.
3. Zapoznać się z podstawowymi poleceniami systemu pomocy Linux.

1. Laboratorium 1 – wskaźniki do funkcji

Celem ćwiczenia jest odświeżenie i utrwalenie wiedzy i umiejętności w zakresie operacji na wskaźnikach i wykorzystania wskaźników do operacji na funkcjach

Zadanie 1. Wykorzystanie wskaźników

Zadanie ma na celu utrwalenie sposobu użycia zmiennych wskaźnikowych.

1. Utworzyć nowy projekt dla języka C w IDE Eclipse typu "Hello world ANSI C Project".
2. Zadeklarować zmienne typu *int, char, unsigned char*.
3. Zadeklarować zmienne wskaźnikowe, które będą przetrzymywały adresy do wcześniej zadeklarowanych zmiennych.
4. Przypisać dowolne wartości zadeklarowanym zmiennym (nie dotyczy zmiennych wskaźnikowych).
5. Przypisać adresy odpowiednich zmiennych typu *int, char, unsigned char* zmiennym wskaźnikowym.
6. Wyświetlić zarezerwowany rozmiar pamięci dla zmiennych typu *int, char, unsigned char* i zmiennych wskaźnikowych a także wartość tych zmiennych.
7. Przypisać poszczególnym zmiennym wartość przez ich wskaźnik.
8. Ponownie wyświetlić rozmiar i wartość zmiennych typu *int, char, unsigned char* i zmiennych wskaźnikowych.
9. W komentarzach (/**/) w kilku zdaniach zapisać wnioski z zadania

Zadanie 2. Wykorzystanie wskaźników na funkcje

Celem zadania jest utrwalenie zasad tworzenia wskaźników na funkcje i ich wykorzystania.

1. Utworzyć nowy projekt dla języka C w IDE Eclipse typu "Hello world ANSI C Project".
2. W pliku z funkcją główną (*main()*) zadeklarować i zdefiniować funkcję *suma()*, która będzie wymagała dwóch argumentów o wartościach całkowitych i będzie wyliczała i zwracała ich sumę.
3. W pliku z funkcją główną (*main()*) zadeklarować i zdefiniować funkcję *iloraz()*, która będzie wymagała dwóch argumentów o wartościach całkowitych i będzie wyliczała i zwracała ich iloraz.
4. Zadeklarować wskaźnik na funkcję tak, aby mógł wskazywać zarówno na funkcję suma jak i na funkcję iloraz.
5. Przypisać wskaźnikowi na funkcję adres funkcji suma().
6. Za pomocą zadeklarowanego wskaźnika wywołać funkcję suma() i wyświetlić zwrócony wynik.
7. Przypisać wskaźnikowi na funkcję adres funkcji iloraz().
8. Za pomocą zadeklarowanego wskaźnika wywołać funkcje iloraz() i wyświetlić zwrócony wynik.
9. Wykonać program w trybie Debug w celu zaobserwowania zmian wartości oraz adresów poszczególnych zmiennych.
10. W komentarzach (/**/) w kilku zdaniach zapisać wnioski z zadania

Zadanie 3. Przekazanie wskaźnika na funkcję do innej funkcji

Zadanie ma na celu pokazanie sposobu wykorzystania wskaźnika do funkcji w innych funkcjach.

1. Do kodu programu z zadania 2, zadeklarować funkcję, która przyjmie jako argumenty dwie liczby całkowite oraz wskaźnik do funkcji o sygnaturze pasującej do funkcji suma i iloraz.

2. Zaimplementować działanie zadeklarowanej funkcji tak, aby wykonywała funkcję przekazaną poprzez wskaźnik w argumencie używając przekazanych do niej liczb typu integer.
3. Wywołać zadeklarowaną funkcję tak, aby wykonała dodawanie i wyświetlić wynik
4. Wywołać zadeklarowaną funkcję tak, aby wykonała iloraz i wyświetlić wynik
5. Spróbować wywołać funkcję podając w miejsce wskaźnika wprost raz nazwę suma, raz iloraz, a raz NULL. Czy program się kompiluje i wykonuje i dlaczego?
6. Wykonać program w trybie Debug w celu zaobserwowania zmian wartości oraz adresów poszczególnych zmiennych.
7. W komentarzach (/**/) w kilku zdaniach zapisać wnioski z zadania

2. Laboratorium 2 – biblioteki dynamiczne

Celem ćwiczenia jest odświeżenie i utrwalenie wiedzy i umiejętności w zakresie tworzenia i wykorzystania bibliotek dynamicznych.

Zadanie 1. Tworzenie biblioteki dynamicznej

Zadanie ma na celu utrwalenie sposobu tworzenia bibliotek dynamicznych.

1. Utworzyć nowy projekt dla języka C w IDE Eclipse typu "Empty Project" o nazwie biblioteka.
2. Wewnątrz utworzonego projektu stworzyć katalog źródłowy o nazwie „src”
3. W katalogu „src” stworzyć plik nagłówkowy *bilblioteka.h* i źródłowy *biblioteka.c.*
4. Zaimplementować funkcje *odejmowanie()* i *iloczyn()* realizujące odpowiednie operacje matematyczne, przyjmujące dwa argumenty typu całkowietego i zwracające wynik.
5. Wykorzystując terminal (konsolę) na podstawie utworzonych plików stworzyć bibliotekę dynamiczną.

Zadanie 2. Wykorzystanie bibliotek dynamicznych – ładowanie/zamykanie podczas wykonywania programu

Celem zadania jest utrwalenie zasad wykorzystania bibliotek dynamicznych.

1. Utworzyć nowy projekt dla języka C w IDE Eclipse typu "Hello world ANSI C Project".
2. W pliku z funkcją główną (*main()*) załadować bibliotekę utworzoną w zadaniu 1.
3. Sprawdzić czy udało się załadowac bibliotekę. W przypadku błędu wyświetlić odpowiedni komunikat.
4. Pobrać wskaźnik na funkcję *odejmowanie()* i za jego pomocą wykonać funkcję oraz wyświetlić otrzymany wynik
5. Pobrać wskaźnik na funkcję *iloczyn()* i za jego pomocą wykonać funkcję oraz wyświetlić otrzymany wynik
6. Zamknąć używaną bibliotekę.
7. Sprawdzic czy biblioteka została zamknięta poprawnie. W przypadku błędu wyświetlić odpowiedni komunikat.
8. Powtórzyć punkty od 2 do 7 wykorzystując drugi typ ładowania.

   W komentarzach (/**/) w kilku zdaniach zapisać wnioski z zadania, uwzględniając czym różnią się obydwa tryby ładowania.

3. Laboratorium 3 – listy wiązane

Celem ćwiczenia jest odświeżenie i utrwalenie wiedzy i umiejętności w zakresie tworzenia i wykorzystania list wiązanych.

Zadanie 1. Listy wiązane

Celem zadania jest wykorzystanie list wiązanych w języku C. Należy napisać program, który będzie buforował nieznaną liczbę par wartości całkowitej i rzeczywistej.

1. Utworzyć nowy projekt dla języka C w IDE Eclipse typu "Hello world ANSI C Project".
2. W pliku nagłówkowym przygotować strukturę, która będzie wykorzystana do przechowywania elementów bufora w dwukierunkowej liście wiązanej.
3. Zadeklarować zmienne pozwalające na powiązanie elementów listy, czyli dla: dowiązania nowego elementu, wskazania poprzedniego elementu i wskazania pierwszego elementu (punktu zakotwiczenia).
4. Przygotować i przypisać wartości dla zmiennych pierwszego elementu na liście.
5. Dodać do listy elementy, tak aby zawierała 10 par wartości.
6. Wyświetlić zawartość poszczególnych elementów listy.
7. Usunąć piąty i siódmy element z listy (pamiętać o zapewnieniu spójności listy).
8. Wyświetlić zawartość poszczególnych elementów listy.

4. Zadanie projektowe

Celem zadania projektowego jest stworzenie aplikacji pozwalającej na stworzenie i wysłanie dowolnego pakietu IP.

Przygotowanie do ćwiczeniach

W ramach przygotowania do ćwiczenia należy zapoznać się z możliwościami programu sendip.

Realizacja ćwiczenia:

1. Każdy ze studentów realizuje projekt samodzielnie. Stos protokołów jakie ma obsługiwać program, dla każdego studenta, określa prowadzący zajęcia:
   a) IPv4 + ICMP
   b) IPv4 + TCP
   c) IPv4 + UDP
   d) IPv6 + ICMPv6
   e) IPv6 + TCP
   f) IPv6 + UDP
2. Program ma obsługiwać pobieranie parametrów z linii poleceń. Dostarczane parametry zależą od stosu protokołów określonego w punkcie 1.
3. Program musi również jako jeden z argumentów przyjmować interfejs za pomocą którego pakiet ma zostać wysłany.
4. Kolejnym argumentem programu ma być liczba pakietów do wysłania.
5. Dla każdego z obsługiwanych protokołów stworzyć bibliotekę pozwalającą na ustawienie poszczególnych pól (np. biblioteka do obsługi IPv4 powinna posiadać funkcję ustawiającą pole wersji, długość nagłówka itp.).
6. Biblioteki te mają być ładowane dynamicznie podczas działania programu.
7. Pola nie podane przez użytkownika powinny przyjmować domyślne wartości zgodne z RFC.
8. Po utworzeniu odpowiedniego pakietu, należy powielić go żądaną liczbę razy i załadować do listy wiązanej.
9. Pakiety z listy należy wysłać do wskazanego interfejsu.

Wymagania:

1. W programie należy wykorzystać wskaźniki na funkcje.
2. Historia prowadzenia i dokumentacja projektu powinna znajdować się na plataformie github.com (lub innej obsługującej system kontroli wersji git).