ДОДАТКОВІ ЛАБОРАТОРНІ РОБОТИ №11*, №12*, №13*, №14*, №15*.

**виконується групами КІ-307, КІ-308 та КІ-309 у разі несвоєчасного виконання основних лабораторних робіт*

ВСТУП ДО МЕРЕЖНОГО ПРОГРАМУВАННЯ: СОКЕТИ БЕРКЛІ ТА ЗМІШАНЕ ПРОГРАМУВАННЯ МОВАМИ GO, C#, JAVA, PYTHON, JS(ДЛЯ NODEJS) ТА АСЕМБЛЕРА

Мета: познайомитися з принципами мережного програмування за допомогою сокетів Берклі та застосувати їх для створення програм, частини яких написані різними мовами програмування.

ТЕОРЕТИЧНІ ВІДОМОСТ МІСТЯТЬСЯ В ДОДАТКОВІЙ ЛАБОРАТОРНІЙ РОБОТІ №8*

КОНТРОЛЬНІ ПИТАННЯ

1. Для чого призначений механізм сокетів Берклі (Berkeley sockets)?
2. Який тип сокетів Берклі використовується для організації мережної взаємодії процесів за протоколом TCP?
3. Для чого використовується системний виклик socket() і яке значення від повертає?
4. Чим відрізняються серверний, комунікаційний та клієнтський сокети?
5. Чим відрізняється взаємодія процесів з використанням сокетів в режимі віртуального каналу (SOCK_STREAM), режимі дейтаграмного зв’язку (SOCK_DGRAM) та у режимі локальної взаємодії (AF_LOCAL)?
6. Який системний виклик ОС Linux використовується для інформування системи про те, що процес серверу планує встановлення віртуальних з'єднань через вказаний сокет із заданою максимальною довжиною черги запитів на встановлення з'єднання?
7. В якому порядку використовуються системні виклики та функції при роботі з серверним сокетом в режимі віртуального каналу (stream sockets)?
8. В якому порядку використовуються системні виклики та функції при роботі з клієнтським сокетом в режимі віртуального каналу (stream sockets)?

ЛІТЕРАТУРА

1. Джордейн Р.Справочник програмиста персональных компъютеров типа IBM PC XT и AT. - М."Финансы и статистика",1992,стор.13-31.
2. Березко Л.О., Троценко В.В. Особливості програмування в турбо-асемблері. - Київ, НМК ВО, 1992.
3. Дао Л. Программирование микропроцессора 8088.Пер. с англ. -М.: "Мир",1988.
4. Абель П.Язык ассемблера для IBM PC и программирования. Пер. з англ.- М.,"Высшая школа",1992.
5. Intel® 64 and IA-32 Architectures Software Developer’s Manual Volume 2 (2A & 2B): Instruction Set Reference, A-Z. – 2011. Режим доступу: http://www.intel.com/content/www/us/en/architecture-and-technology/64-ia-32-architectures-software-developer-vol-2a-2b-instruction-set-a-z-manual.html
6. Maurice Herlihy, Nir Shavit, Victor Luchangco, Michael Spear, The Art of Multiprocessor Programming, 2nd Edition, Morgan Kaufmann, 2020. – 576 p.
7. Peter Pacheco, Matthew Malensek, An Introduction to Parallel Programming, 2nd Edition,

Morgan Kaufmann, 2021. – 450 p.
8. Bertil Schmidt Jorge Gonzalez-Dominguez Christian Hundt Moritz Schlarb, Parallel Programming: Concepts and Practice, Morgan Kaufmann, 2017. – 416 p.
9. Programming Models for Parallel Computing, ed. by Pavan Balaji, The MIT Press, 2015. - 488 p.
10. Michael McCool, James Reinders, Arch Robison, Structured Parallel Programming: Patterns for Efficient Computation, Morgan Kaufmann, 2012. – 432 p.
11. Thomas Rauber, Gudula Rünger, Parallel Programming For Multicore and Cluster Systems, Springer, 2010. – 455 p.
12. Fayez Gebali, Algorithms and Parallel Computing, John Wiley & Sons, 2011. – 365 p.
13. Victor Alessandrini, Shared memory application programming, Morgan Kaufmann, 2016. – 528 p.
14. W. Richard Stevens, UNIX Network Programming, Volume 2: Interprocess Communications, 2nd ed., Prentice Hall, 1998. – 558 p.
15. W. Richard Stevens, UNIX Network Programming (Volume 1): Networking APIs: Sockets and XTI, Prentice Hall, 1998. – 1009 p.
16. Michael Kerrisk, The Linux Programming Interface: A Linux and UNIX System Programming Handbook, 1st Edition, No Starch Press, 2010. – 1553 p.
17. W. Stevens, Stephen Rago, Advanced Programming in the UNIX Environment, 3rd Edition, Addison-Wesley Professional, 2013. – 1032 p.
18. Robert Love, Linux System Programming, 2nd ed., O'Reilly Media, 2013. – 456 p.
19. Kaiwan N. Billimoria, Hands-On System Programming with Linux, Packt Publishing, 2018. - 794 p.
20. Ulrich Drepper, Ingo Molnar, The Native POSIX Thread Library for Linux, White paper, Red Hat, Inc., 2005.
21. Rohit Chandra et al. Parallel Programming in OpenMP, Academic Press, Morgan Kaufmann Publishers, 2001. – 249 p.
22. Barbara Chapman, Gabriele Jost, Ruud van der Pas, Using OpenMP: Portable Shared Memory Parallel Programming, The MIT Press, 2008. – 353 p.
23. Ruud Van Der Pas, Eric Stotzer, Christian Terboven, Using OpenMP-The Next Step: Affinity, Accelerators, Tasking, and SIMD, The MIT Press, 2017. – 392 p.
24. Timothy G. Mattson, Yun (Helen) He, Alice E. Koniges, The OpenMP Common Core, The MIT Press, 2019. – 320 p.
25. OpenMP Application Programming Interface Specification Version 5.2, November 2021. – 649 p.
26. James Reinders, Intel Threading Building Blocks: Outfitting C++ for Multi-core processor Parallelism, O'Reilly Media, 2007. – 336 p.
27. Michael Voss, Rafael Asenjo, James Reinders, Pro TBB: C++ Parallel Programming with Threading Building Blocks, Apress, 2019. – 820 p.
28. Hagit Attiya, Jennifer Welch, Distributed Computing: Fundamentals, Simulations and Advanced Topics, 2 ed., John Wiley & Sons, 2004. – 414 p.
29. George Em Karniadakis, Robert M. Kirby II, Parallel Scientific Computing in C++ and MPI, Cambridge University Press, 2003. - 696 p.

ЗАВДАННЯ

1. Створити *.exe програму, яка реалізовує обчислення, заданого варіантом виразу. Програма повинна складатися з двох модулів:
   головний модуль – створюється мовою С(або С++) і має забезпечити html-інтерфейс(для вводу і виводу даних) за допомогою http-протоколу та виклик асемблерної процедури для обчислення виразу;
   модуль безпосередніх обчислень – здійснює всі необхідні арифметичні дії з використанням математичного співпроцесора.
2. Переконатися у правильності роботи кожного модуля зокрема та програми загалом.
3. Скласти звіт про виконану роботу з приведенням тексту програми та коментарів до неї.
4. Дати відповідь на контрольні запитання.

Таблиця 7.1.

Варіанти завдань

(відповідають варіантам додаткової лабораторної роботи №6)

№	Вираз	K
1.	$X=A_4+B_2*C_1-D_2/E_1+K$	1254021
2.	$X=A_4/B_2+C_3-D_1*E_1-K$	202
3.	$X=K-B_4+C_2/D_1-E_1*F_2$	37788663
4.	$X=A_1*(B_2+C_1)-D_1/E_2+K$	45694
5.	$X=A_2*B_2-A_2*C_1-D_1/E_2+K$	505
6.	$X=K+B_2/C_1-D_2*F_2-E_1$	6DD02316
7.	$X=A_4/B_2-C_1*(D_1+E_2-K)$	717
8.	$X=A_2-B_1+K-D_2/E_1+F_1*B_2$	88
9.	$X=A_1*B_2-A_1*C_2+D_2/(E_1+K)$	29
10.	$X=A_2-B_1/C_2+K+E_2*F_1$	2310
11.	$X=(A_2-B_1-K)*D_1+E_4/F_2$	311
12.	$X=K+B_1/C_2-D_2*F_2-E_1$	7055E0AC
13.	$X=A_2/B_1+C_1*(D_1+E_2-K)$	2513
14.	$X=A_2-B_1-K-D_2/E_1+F_1*B_1$	614
15.	$X=A_2+(B_1*C_2)-D_4/E_2+K$	4569600F
16.	$X=A_1/B_2+C_3-D_1*E_1+K$	616
17.	$X=A_1-K+C_1/D_2-E_1*F_1$	1017
18.	$X=A_1*(B_2-C_1)+D_2/E_1+K$	56987018
19.	$X=A_2*B_2+A_2*C_1-D_2/E_1+K$	4019
20.	$X=K+B_1/C_2-D_1*F_1-E_2$	18932020
21.	$X=A_1/B_2+C_1*(D_2-E_1+K)$	21
22.	$X=K-B_2-C_1-D_2/E_1+F_2*B_1$	45781022

23.	$X=A_2*B_2-C_1+D_1/E_2+K$	7AA02023
24.	$X=K-B_2/C_1+D_1+E_2*F_2$	74569024
25.	$X=(K-B_2-C_1)*D_1+E_4/F_2$	2B05025
26.	$X=A_2+K+C_2/D_1-E_1*F_1$	6C26
27.	$X=A_2*B_1+C_1/(K-E_1*F_1)$	A77627
28.	$X=K+B_1/C_2+D_2-E_2/F_1$	3FF28
29.	$X=K-B_1*C_1+D_2-F_2/E_1$	12A0C029
30.	$X=K+B_2-D_2/C_1+E_1*F_2$	25630

A, B, C, D, E, F - дійсні числа, де індекс визнає точність внутрішнього подання (1 – одинарної, 2, 3, 4 – подвійної точності). Константу К подано у 16-му форматі.

ПРИКЛАД КОДУ ДЛЯ ДОДАТКОВОЇ ЛАБОРАТОРНОЇ РОБОТИ №11 (GO)

Програма з взаємодією Go-ASM, що виконує завдання згідно варіанту №30 розміщена за посилапнням:

https://github.com/KozakNazar/srcserver__go_asm/tree/master/LLNW_extended .

ПРИКЛАД КОДУ ДЛЯ ДОДАТКОВОЇ ЛАБОРАТОРНОЇ РОБОТИ №12 (C#)

Програма з взаємодією C#-ASM(x86), що виконує завдання згідно варіанту №30 розміщена за посилапнням:

https://github.com/KozakNazar/srcserver__cs_asm/tree/master/LLNW_extended .

ПРИКЛАД КОДУ ДЛЯ ДОДАТКОВОЇ ЛАБОРАТОРНОЇ РОБОТИ №13 (JAVA)

Програма з взаємодією Java-ASM(x86), що виконує завдання згідно варіанту №30 розміщена за посилапнням:

https://github.com/KozakNazar/srcserver__java_asm/tree/master/LLNW_extended .

ПРИКЛАД КОДУ ДЛЯ ДОДАТКОВОЇ ЛАБОРАТОРНОЇ РОБОТИ №14 (PYTHON)

Програма з взаємодією Python-ASM(x86), що виконує завдання згідно варіанту №30 розміщена за посилапнням:

https://github.com/KozakNazar/srcserver__python_asm/tree/master/LLNW_extended .

ПРИКЛАД КОДУ ДЛЯ ДОДАТКОВОЇ ЛАБОРАТОРНОЇ РОБОТИ №15 (JS ДЛЯ NODEJS)

Програма з взаємодією JS(NodeJs)-ASM(x86), що виконує завдання згідно варіанту №30 розміщена за посилапнням:

https://github.com/KozakNazar/srcserver__nodejs_asm/tree/master/ILNW_extended .