UNIX SYSTEMS PROGRAMMING AND COMPILER DESIGN LABORATORY

Subject Code: 10CSL68 **I.A. Marks: 25**
Hours/Week : 03 **Exam Hours: 03**
Total Hours : 42 **Exam Marks: 50**

List of Experiments for USP: Design, develop, and execute the following programs:

1. Write a C/C++ POSIX compliant program to check the following limits:
   - **i.** No. of clock ticks
   - **ii.** Max. no. of child processes
   - **iii.** Max. path length
   - **iv.** Max. no. of characters in a file name
   - **v.** Max. no. of open files/process.

2. Write a C/C++ POSIX compliant program that prints the POSIX defined configuration options supported on any given system using feature test macros.

3. Consider the last 100 bytes as a region. Write a C/C++ program to check whether the region is locked or not. If the region is locked, print pid of the process which has locked. If the region is not locked, lock the region with an exclusive lock, read the last 50 bytes and unlock the region.

4. Write a C/C++ program which demonstrates interprocess communication between a reader process and a writer process. Use mkfifo, open, read, write and close APIs in your program.

5. a) Write a C/C++ program that outputs the contents of its Environment list
   b) Write a C / C++ program to emulate the unix **ln** command

6. Write a C/C++ program to illustrate the race condition.

7. Write a C/C++ program that creates a zombie and then calls system to execute the **ps** command to verify that the process is zombie.

8. Write a C/C++ program to avoid zombie process by forking twice.

9. Write a C/C++ program to implement the **system** function.

10. Write a C/C++ program to set up a real-time clock interval timer using the **alarm** API.

List of Experiments for Compiler Design: Design, develop, and execute the following programs.

11. Write a C program to implement the syntax-directed definition of "if E then S1" and "if E then S1 else S2". (Refer Fig. 8.23 in the text book prescribed for 06CS62 Compiler Design, Alfred V Aho, Ravi Sethi, and Jeffrey D Ullman: Compilers- Principles, Techniques and Tools, 2nd Edition, Pearson Education, 2007).

12. Write a yacc program that accepts a regular expression as input and produce its parse tree as output.

Note: In the examination *each* student picks one question from the lot of *ll* 12 questions.