Uva Wellassa University, Sri Lanka
Faculty of Science & Technology
2nd Semester Examination August/September 2011
CST 309-3 System Level Programming

Uva Wellassa University

This is an open book examination

Time Duration : **Three(03) hours**
Number of Questions : Four(04)
Number of Pages : Three(03)

Instructions to Candidates :

Answer **All** questions. All questions carry 25 marks.
Create a single compressed file including all the answers, the name of the compressed file should be same as your examination number (Ex: EX08001.zip) and upload to the CMS.

Q1. Shell scripting **[25 marks]**

a) Write a shell script **question1.sh** which will print the location of the current working directory, username, date and time.

After executing the script it should display the output as follows.

The script was run by exam000 on the 26/Aug/2011 at 10:10:20 AM
The working directory was /home/exam000/Desktop/question1

b) Modify question1.sh to create a directory called JSP inside the current working directory.

Assume, the current working directory is /home/exam000/Desktop/question1 (exam000 is the username) and you need to create the JSP directory as /home/exam000/Desktop/question1/JSP

c) Modify the above shell script to copy all the "*.html" files in the current working directory to the JSP directory.

Assume, a.html, b.html and c.html files are available in the particular directory.

d) Modify the above shell script to convert all the "*.html" files in the JSP directory to "*.jsp" files.

When you finish executing your script, you should have a.html , b.html , c.html in the current directory and a.jsp , b.jsp , c.jsp in the JSP directory and you should also have a log file named html2jsp.log with the following format.

#
This is a log file of html2jsp script.
#

The script was run by exam000 on the 26/Aug/2011 at 10:10:20 AM
The working directory was /home/exam000/Desktop/question1

There were 3 files renamed and they are:

a.html -> a.jsp
b.html -> b.jsp
c.html -> c.jsp

Q2. C programing – Control flows and Recursive methods **[25 marks]**

a) Write a program to generate Fibonacci series depends on the user input. The user will enter some number, which we will call n. Your program should use a **for loop** to generate the Fibonacci numbers up to n, and then use a **while loop** to generate the numbers from n back down to zero.

In mathematical terms, the sequence F(n) of Fibonacci numbers is defined by,

$F(n) = F(n+1) + F(n+2)$

where F(0)=0 and F(1)=1 and the sequence of Fibonacci series is 0,1,1,2,3,5,8,13....

b) A palindrome is a string that is spelled the same way forwards and backwards. Write a recursive method named palindrome that returns an integer 1 if the string is a palindrome and 0 otherwise. Write a main method to test the program. Your program should ignore the spaces.

Some examples of palindromes are: "noon", “god saw I was dog”, and
"a man a plan a canal panama".

Q3. C programing – Pointers and File handling **[25 marks]**

a) Write a program to implement **bubble sorting algorithm** (Exchange two adjacent elements if they are out of order. Repeat until array is sorted.). In which user should enter the elements to be sorted and maximum number of elements that user can provide should be 100.

b) Write a program to compare two files specified by the user, displaying a message indicating whether the files are identical or different and the message should be directed to another file.

Q4. Processes and System calls **[25 marks]**

The purpose of **fork()** is to create a new process, which becomes the child process of the caller. It takes no arguments and returns a process ID. After a new child process is created, both processes will execute the next instruction following the fork() system call. Therefore, we have to distinguish the parent from the child. This can be done by testing the returned value of fork() as follows.

- fork() returns a negative value if the creation of a child process was unsuccessful.
- fork() returns a zero to the newly created child process.
- fork() returns a positive value (the process ID of the child process) to the parent. The returned process ID is of type pid_t defined in sys/types.h. Normally, the process ID is an integer. Moreover, a process can use function getpid() to retrieve the process ID assigned to this process.

Following example does not distinguish the parent process from the child process.

```
#include <stdio.h>
#include <string.h>
#include <sys/types.h>
#define MAX_COUNT 100
void main(void)
{
   pid_t pid;
   int   i;

   fork();
   pid = getpid();
   for (i = 1; i <= MAX_COUNT; i++) {
      printf("This line is from pid %d, value = %d\n", pid, i);
    }
}
```

Write a C program which distinguishes the parent from the child.

48