RTOS Lab Assignment – Set 1

Write and execute C programs to do the following tasks.

1. Get ten numbers from user. Store them in an array. Using functions, find sum, average and product of the numbers. Print the sum, average and product in the main function.
2. Modify the above program to continue to accept the input from the user until the user enters '999'. As the user inputs the number, the sum, average and product should be displayed in the main function. Use a static variable in the program.
3. Modify the program above to accept the numbers through the command line arguments.
4. Modify the program in Qn.2 to read the numbers from a file.
5. Create a telephone directory, with the first name, last name and telephone number. Provide the facility to order the entries of the directory according to the alphabetical ordering of the first name or based on the telephone number. Get the data from a file and use switch –case in your program. Provide user option to view entries ordered based on name or number.
6. Compare two dates and display the latter one. Consider the date as consisting of day, month and year. Get the dates through user inputs through console. Use structures and 'typedef' in this program.
7. Count the number of characters in a given string without making use of the string library function. Use 'for loop'. Get the input using 'fgets'.
8. Read the text stored in a file. Count the number of words and lines in the file. Use functions.
9. Get an input string and an option value from the command line. Indicate the end of string using "FFF". The option value of 0 or 1 is expected to follow in the command line after "FFF". The program should check for a match to "MCIS" in the given input if option value is 0. If the option value is 1, the program should check for "SOIS". If it is not so, the program should display the count of words excluding "FFF" I the input string. Write a 'compare' function to compare the two words.
10. Read a text file called input.txt and segregate the even and odd words into two different files names even.txt and odd.txt.

RTOS Lab Assignment – Set 2

Use parent process – child process relationship in the following programs, use files to pass data between parent and child processes:

1. Get ten numbers from user. Store them in an array. Using functions, find sum, average and product of the numbers. Obtain the sum value from the child, calculate and print the average in the parent process. Use a child process to calculate and print the sum and product of the numbers.
2. Modify the above program to continue to accept the input from the user until the user enters '999'. As the user inputs the number, the sum, average and product should be displayed in the parent process. Use a child process to calculate the sum and product in the program.

3. Create a telephone directory, with the first name, last name and telephone number. Provide the facility to order the entries of the directory according to the alphabetical ordering of the first name or based on the telephone number. Get the data from a file and use child process to order the entries based on number or name.
4. Count the number of characters in a given string without making use of the string library function. Use a child process to count the number of vowels and display to the console.
5. The main program reads a text stored in a file. Let the parent process count the number of words in the text and display on console. Create a child process to count the number of characters and pass on this value to the parent process to display on to the console.
6. Get an input string and an option value from the command line. Indicate the end of string using “FFF”. The option value of 0 or 1 is expected to follow in the command line after “FFF”. The parent process should check for a match to “MCIS” in the given input if option value is 0. If the option value is 1, the program should check for “SOIS”. Create a child process to count of words excluding “FFF” in the input string. Write a ‘compare’ function to compare the two words.
7. Read a text file called input.txt. Let the parent process segregate the even and odd words into two different files names even.txt and odd.txt. Create a child process to list the words from even.txt onto the console. Let the parent process create another child process to list the words from odd.txt. Let the parent process display “I am done” once the child process finish displaying the words onto the console.

Repeat questions 1-7 with data communication between parent –child processes through pipes.

Repeat questions 1-7 with the processes getting input data from shared memory.

Repeat questions 1-7 with the multiple threads