Lab 2

Interaction With the Execution Environment

When the operating system executes a program, it automatically provides certain facilities that help the program communicate with the operating system and the user. We consider the the environment in which the programs, how they can use that environment to gain information about the operating system conditions, and how users of the program can alter their behavior.

The Argument list

We run a program from a shell prompt by typing the name of the program. Optionally, we can supply the additional information to the program by typing one or more words after the program name, separated by spaces. These are called the command line arguments or argument list.

Ex.2.1. The following program demonstrates how to use *argc* and *argv*.

a*rglist.c*

```
#include < stdio.h>

int main ( int argc, char* argv[])
{
        printf (“ The name of this program is:  '%s'. \n”, argv[0]+2);
        printf (“ This program is invoked with %d arguments \n”, argc – 1);

        if (argc>1)
        {
                int i;
                printf(“ The arguments are :\n”);
                for (i = 1; i < argc; ++i)
                        printf(“%s \n”, argv[i]);
        }
return 0;
}
```

The Environment

Linux provides each running program with an environment. The environment is a collection of variable/value pairs. Both environment variable names and their values are character strings. By convention, environment variable names are spelled in all capital letters.

Examples:

USER: contains user name.

HOME: contains the path to the home directory.

PATH: contains the colon-separated list of directories through which Linux search for the command.

'printenv'- To print the current environment variables .

Ex. 2.2. Printing the environment variables.

showenv.c

```
#include<stdio.h>
extern char **environ;
int main()
{
        char **env = environ;
        while(*env)
                {
                printf(" %s \n", *env);
                env++;
                }
        return 0;
}
```

Note: After saving the above program, issue the following command to execute as command.

```
chmod a+x <filename>
PATH=$PATH:.
$ <filename>
```

Assignment #L2:

1. Run the above programs at lest three times (Ex. 2.1 and Ex. 2.2) and analyze the output?
2. Write the program which create the new file – makefile (equivalent to 'touch' command).