ROLL NO. _____

Code: AC71/AT71 Subject: UNIX SYSTEMS PROGRAMS

AMIETE – CS/IT (NEW SCHEME)

Time: 3 Hours **DECEMBER 2011** **Max. Marks: 100**

NOTE: There are 9 Questions in all.

- **Please write your Roll No. at the space provided on each page immediately after receiving the Question Paper.**
- **Question 1 is compulsory and carries 20 marks. Answer to Q.1 must be written in the space provided for it in the answer book supplied and nowhere else.**
- **The answer sheet for the Q.1 will be collected by the invigilator after 45 Minutes of the commencement of the examination.**
- **Out of the remaining EIGHT Questions, answer any FIVE Questions. Each question carries 16 marks.**
- **Any required data not explicitly given, may be suitably assumed and stated.**

Q.1 Choose the correct or the best alternative in the following: (2×10)

a. If the process time is stored as a signed 32-bit integer, and if the system counts 100 ticks per second, after how many days (approximately) will the value overflow?

(A) 248 **(B)** 512
(C) 128 **(D)** None of the above

b. Data is read from an open file with the read function. If the read is successful, the number of bytes read is returned. If the end of file is encountered, _____ is returned.

(A) 1 **(B)** -1
(C) 0 **(D)** None of the above

c. *tempnam* restrict the *prefix* to _____ characters

(A) 6 **(B)** 4
(C) 5 **(D)** None of the above

d. The shells process their command lines _____

(A) from right to left **(B)** from left to right
(C) Both **(A)** & **(B)** **(D)** None of the above

e. On an Intel x86 system under both FreeBSD and Linux, if we execute the program that prints "hello world" and do not call *exit* or *return*, the termination status of the program, which we can examine with the shell is _____

(A) 15 **(B)** 14
(C) 12 **(D)** 13

f. File descriptors 2, 1 and 0 can be replaced in POSIX-compliant applications with symbolic constants:

(A) STDIN_FILENO, STDOUT_FILENO, STDERR_FILENO respectively
(B) STDOUT_FILENO, STDIN_FILENO, STDERR_FILENO respectively
(C) STDERR_FILENO, STDIN_FILENO, STDOUT_FILENO respectively
(D) STDERR_FILENO, STDOUT_FILENO, STDIN_FILENO respectively

g. In UNIX system terminology, a process that has terminated, but whose parent has not yet waited for it, is called

(A) abort **(B)** quit
(C) null **(D)** zombie

h. The fastest form of IPC is

(A) shared memory **(B)** pipes
(C) named pipes **(D)** semaphores

i. An existing process can create a new one by calling the _____ function.

(A) fork **(B)** exit
(C) wait **(D)** invoke

j. The _____ function is the catchall for I/O operations.

(A) fsync **(B)** fcntl
(C) ioctl **(D)** fdatasync

Answer any FIVE Questions out of EIGHT Questions.
Each question carries 16 marks.

Q.2 a. If the calendar time is stored as a signed 32-bit integer, in what year will it overflow? What ways can be used to extend the overflow point? Are they compatible with existing applications? **(8)**

b. The Bourne shell, Bourne-again shell, and Korn shell notation

digit1 >&*digit2*

means to redirect descriptor *digit1* to the same file as descriptor *digit2*. What is the difference between the two commands

./a.out > outfile 2>&1
./a.out 2>&1 > outfile **(8)**

Q.3 a. What happens if the file mode creation mask is set to 777 (octal)? Verify the results using your shell's *umask* command. **(4)**

b. Each process has a root directory that is used for resolution of absolute pathnames. This root directory can be changed with *chroot* function. When might this function be useful? **(8)**

c. A file size of 0 is valid for a regular file. The *st_size* field is defined for directories and symbolic links. Should we ever see a file size of 0 for a directory or a symbolic link? Explain. **(4)**

Q.4 a. Why does *tempnam* restrict the *prefix* to five characters? **(3)**

b. List and briefly explain three basic functions of data files. **(9)**

c. Name and explain meaning of the various fields in a group file. **(4)**

Q.5 a. Describe three reasons for use of interpreter files. **(6)**

b. What is a *zombie*? Write a program that creates a *zombie*. **(10)**

Q.6 a. Is there any way for a function that is called by main to examine the command-line arguments without
(i) passing *argc* and *argv* as arguments from main to the function or
(ii) having main copy *argc* and *argv* into global variables **(5)**

b. Some Unix system implementations purposely arrange that, when a program is executed, location 0 in the data segment is not accessible. Why? **(4)**

c. Why are the logout records written by the init process? Is this handled the same way for a network login? **(7)**

Q.7 a. Briefly explain following signals:
(i) SIGABRT (ii) SIGALRM
(iii) SIGKILL (iv) SIGIOT
(v) SIGPIPE **(10)**

b. Draw and explain pictures of the stack frames when we run the following program: **(6)**

```
#include        < apue.h >

unsigned int    sleep2 (unsigned int);
static void     sig_int (int);
int
main (void)
{
                unsigned int    unslept ;
                if      (signal (SIGINT,  sig_int) = = SIG_ERR)
                err_sys ("signal (SIGINT) error");
                unslept = sleep2 (5) ;
                printf ("sleep2 returned: %u \ n", unslept) ;
                exit (0) ;
```

```
}
static void
sig_int (int signo)
{
        int             i,  j ;
        volatile int    k ;

        printf ("\nsig_int starting\n");
        for     ( i = 0 ; i  < 300000;  i++ )
        for     ( j = 0 ; j  < 4000 ; j++ )
                k += i * j ;
        printf ( "sig_int  finished \ n") ;
}
```

Q.8 a. The PARODD flag in the *c_cflag* field allows us to specify even or odd parity. The BSD tip program, however, also allows the parity bit to be 0 or 1. How does it do this? **(8)**

b. If your system's *stty*(1) command outputs the MIN and TIME values, do the following exercise: Log in to the system twice and start the *vi* editor from one login. Use the sty command from your other login to determine what values *vi* sets MIN and TIME to. Explain the output. **(8)**

Q.9 a. The rationale for POSIX.1 gives as one of the reasons for adding the *waitpid* function that most pre-POSIX.1 systems can't handle the following: **(6)**

```
if  ((fp = popen("/bin/true", "r")) == NULL)
    . . .
if ((rc = system("sleep 100")) == -1)
   . . .
if (plclose(fp) == -1)
   . . .
```

What happens in this code if *waitpid* isn't available and *wait* is used instead?

b. Explain how *select* and *poll* handle an input descriptor that is a pipe, when the pipe is closed. **(10)**