Index number: .....

UNIVERSITY OF RUHUNA

Faculty of Engineering

End-Semester 6 Examination in Engineering: January 2022

dule Number: EE6206

Module Name: Operating Systems and Programming (N/C) –
Part I (MCQ)

[Forty Minutes only]

[This paper consists of 20 MCQs. Answer all questions, each question carries 0.5 marks.
Part I Will be collected at the end of forty minutes.
Write the index number in all pages.
Select only one correct answer for each of the questions.
Underline or cross the correct answer for each question.]

Which of the following is not a way of creating an empty text file Doc1.txt?

(i) Type `cat > Doc1.txt` in console. Then, press Enter. Then, press `Ctrl + D`.
(ii) Type `gedit Doc1.txt in console`. Then, press `Ctrl + S`.
(iii) Type `touch Doc1.txt` in console.
(iv) Type `gedit Doc1.txt in console`. Then, press `Ctrl + C`.
(v) Type `cat > Doc1.txt` in console. Then, press Enter. Then, press `Ctrl + C`.

What is the correct Linux console command to append the content of doc1.txt to an existing doc2.txt?

(i) `cat doc1.txt doc2.txt`
(ii) `cat doc2.txt doc1.txt`
(iii) `cat doc2.txt >> doc1.txt`
(iv) `cat doc1.txt > doc2.txt`
(v) `cat doc1.txt >> doc2.txt`
(vi) `cat doc2.txt > doc1.txt`

The authorization of the file txt1.txt is altered as follows. Give only the 'execute privilege' to a user who is not in user group and remove all other privileges. Select the correct set of commands to obtain this task.

(i) chmod o=x txt1.txt
(ii) chmod g+x txt1.txt
chmod g-r txt1.txt
chmod g-w txt1.txt
(iii) chmod u=x txt1.txt
(iv) chmod u+x txt1.txt

chmod u-r txt1.txt
chmod u-w txt1.txt
(v) chmod g=x txt1.txt

Q4 Select the statement which will cause the Operating System to shut-down?

(i) gnome session quit
(ii) sudo init 6
(iii) sudo init 0
(iv) sudo shutdown -r now
(v) sudo reboot

Q5 Consider the following code segment.

```
void ascending_swap(double *p1, double *p2);
int main(void)
{
      double values[10];
      for(int i=0;i<10;i++)
      {
            printf("\nEnter number %d: ", (i+1));
            scanf("%lf",&values[i]);
      }
      ascending_swap(values[1], values[3]); //Line'10
      return(0);
}
void ascending_swap(double *p1,double *p2) //Line 13
{
      if(*p1 > *p2) //Line 15
      {
            double temp;
            temp = *p2;
            *p2 = *p1;
            *p1 = temp;
      }
      else{}
}
```

Which of the following statement is true regarding the above code segment?

(i) Line 10 is not erroneous. *p1 in line 13, 15 tells that double value at the address pointed by pointer p1, p1 is a pointer of type double respectively.

(ii) Line 10 is erroneous. *p1 in line 13, 15 tells that p1 is a pointer of type double, double value at the address pointed by pointer p1 respectively.

(iii) Line 10 is not erroneous. *p1 in line 13, 15 tells that p1 is a pointer of type double, double value at the address pointed by pointer p1 respectively.

(iv) Line 10 is erroneous. *p1 in line 13, 15 tells that double value at the address pointed by pointer p1, p1 is a pointer of type double respectively.

Consider the following code segment.

```
union myunion
{
    char char_value;
    int int_value;
    double double_value;
};

struct mystruct
{
    char char_value;
    int int_value;
    double double_value;
};
struct mystruct initialize(char * pt1, int * pt2, double * pt3);
int main()
{
    char usr_char = "A"; int usr_int = 60; double usr_double 12.50;
    union myunion uni1;
    uni1.char_value = usr_char;
    uni1.int_value = usr_int;
    uni1.double_value = usr_double;
    struct mystruct str1 = initialize(&usr_char, &usr_int, &usr_double);
    printf("\n For Union: character is %c, integer is %d and double is %lf",
    uni1.char_value, uni1.int_value, uni1.double_value);
    printf("\n For structure: character entered is %c, integer is %d and double
    is %lf", str1.char_value, str1.int_value, str1.double_value);
    return(0);
}
struct mystruct initialize(char * pt1, int * pt2, double * pt3)
{
    struct mystruct str1;
    str1.char_value = *pt1;
    str1.int_value = *pt2;
    str1.double_value = *pt3;
    return str1;
}
```

Index number: .....

Which of the following statement is true regarding the above code segment?

(i) For the union: The terminal screen will show **double_value** as 12.50 while other two variables will not be shown as user specified values. For the structure: The terminal screen will show **double_value** as 12.50, **int_value** as 60, **char_value** as AB

(ii) For the union: The terminal screen will show **int_value** as 60 while other two variables will not be shown as user specified values. For the structure: The terminal screen will show **double_value** as 12.50 while other two variables will not be shown as user specified values.

(iii) For the union: The terminal screen will show **char_value** as AB while other two variables will not be shown as user specified values. For the structure: The terminal screen will show **double_value** as 12.50 while other two variables will not be shown as user specified values.

(iv) For the union: The terminal screen will show **double_value** as 12.50, **int_value** as 60, **char_value** as AB. For the structure: The terminal screen will show **double_value** as 12.50, **int_value** as 60, **char_value** as AB

Q7 Consider the given pseudo code.

```
Return type thread_function1(arguments)
{
      Lock the mutex1;
      statements set 1;
      pthread_cond_signal(&cond2);
      pthread_cond_wait(&cond1,&mutex1);
      statements set 2;
      unlock the mutex1;
      exit the thread;
}
```

Which statement is false out of the following?

(i) The thread having start function as thread_function1 will have to wait until the owner thread of the mutex1 relinquish the lock before executing statements set1.

(ii) When thread_function1 is executing, some other thread sleeping based on a condition variable pointed by cond2 will wake up from sleep and lock the mutex.

(iii) Thread which calls thread_function1 will unlock the mutex temporarily until the condition variable pointed by cond2 is signaled by another thread.

(iv) If no other thread ever signals on condition variable pointed by cond1; after the thread calling thread_function1 goes to sleep; statements set 2 will never be executed.

Index number: .....

What is false about *fork( )* function?

(i) After calling this function, the new process will have a separate copy of the virtual memory address space for stack, heap and data segments.
(ii) It is not known whether the child or parent execute after a *fork( )* call.
(iii) *fork( )* function returns the process identifier of the child process inside the parent process.
(iv) It returns -1 on error and new process created will execute program text stored in different address spaces of the virtual memories of the parent and child processes.

What is true about waitpid(agr1,NULL,0) function?

(i) If arg1 is set to -1, then parent will wait until all of its children terminates. Termination occurs when after all children calls exit( ) function.
(ii) If arg1 is the process identifier of the child process, then parent will wait until that child terminates. Termination of another child will not stop the waiting of parent process.
(iii) If arg1 is the process identifier of the parent process, then child will wait until parent terminates.
(iv) This function is declared in <sys/ipc.h> file

What is true out of the following statements?

(i) The tendency of a process to refer to adjacent memory addresses of an already accessed process is known as temporal locality of reference.
(ii) The function of a page table is to map virtual memory addresses to pages in primary memory.
(iii) All of the virtual memory addresses of a particular process are stored in primary memory at a given instance.
(iv) Pointers to Machine language instructions reside in Heap segment of virtual memory of a process.

What is false out of the following statements?

(i) sleep function can suspend the execution of the process which sleep(t1) was called for t1 seconds.
(ii) When the exit( ) function is called, only the called process and all its associated threads will terminate.
(iii) When the exit( ) function is called by the parent, all its child processes and all of those child processes' associated threads will terminate.

(iv) When a process calls a function, the function's variables are stored as stack frames in virtual memory.

Q12 What is false about pipes?

(i) Pipes are stored in Kernel memory and used for communication between processes.
(ii) Function fprintf( ) called in a process can be used to send bytes to the write end of the pipe.
(iii) Function read( ) called in a process can be used to output data from the pipe to the process using read end of the pipe.
(iv) When a process reads from a pipe, the read data is not removed from the pipe and is accessible for another process to read.

Q13 What is true about the following statements?

```
int fd[2];
int x = pipe(fd);
fork( );
```

(i) fd[0] is associated with write end of the file.
(ii) fd[1] is associated with read end of the file.
(iii) If x is zero, a pipe is created.
(iv) Parent process and child process after calling *fork( )* will share two file descriptors fd with the pipe.

Q14 Select the correct order of the header files that defines each of the functions *fork( ), exit( ), pipe( ), printf( ), getpid( )* respectively?

(i) <unistd.h>, <stdlib.h>, <unistd.h>, <stdio.h>, <unistd.h>
(ii) <stdlib.h>, <unistd.h>, <stdio.h>, <unistd.h>, <errno.h>
(iii) <unistd.h>, <sys/wait.h>, <unistd.h>, <stdio.h>, <errno.h>
(iv) <unistd.h>, <stdio.h>, <unistd.h>, <stdlib.h>, <unistd.h>

Q15 What is true about int msgget(key_t key1, int msgflag)

(i) msgget( ) resembles the function write( ) in File I/O.
(ii) If IPC_EXCL is passed to msgflag and a message queue created using key1 already exists, msgget( ) will return -1.
(iii) passing 0666 to msgflag will cause the user group to have read only privileges.
(iv) Passing the phrase IPC_CREAT will cause the msgget( )to return an error if a message queue produced using the given key exists and create a message queue if it does not exist.

Index number: .....

A program written in C contains the following statements.

```
int fp1=open("doc1.txt",O_RDWR|O_CREAT|O_TRUNC,S_IRUSR|S_IWUSR|S_IRGRP|
S_IWGRP|S_IROTH|S_IWOTH);
printf("%d", fp1);
```

Select the false statement regarding the given statement.

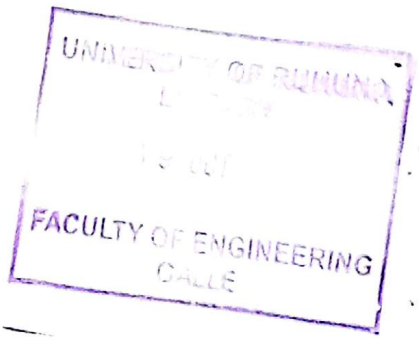
(i) If doc1.txt does not exist; value of fp1 will be -1
(ii) If doc1.txt exists; the content will be deleted
(iii) All user, user group and other users can read from and write to doc1.txt
(iv) If doc1.txt does not exist; a new file will be created and the value of fp1 will be positive.
(v) If doc1.txt cannot be opened; value of fp1 will be -1

A text file txt1.txt contains the following text.

"Hello this is operating system programming end semester examination"

Following set of instructions are run inside a program.

```
int fd1;
fd1 = open("txt1.txt", O_RDONLY,0666);
char buf1[10];
lseek(fd1,14,SEEK_SET);
lseek(fd1,10,SEEK_CUR);
read(fd1,buf1,10);
```



The content of the buffer buf1 is given by;

(i) "operating "
(ii) "system pro"
(iii) "Hello this "
(iv) "programmin"

What is true out of the following?

(i) sem_wait( ) and sem_post( ) called by a thread function has equivalent behavior compared to pthread_mutex_lock( ) and pthread_mutex_unlock( ) respectively.
(ii) Semaphores are stored in user space so that they can be used for process synchronization.
(iii) Mutually exclusive locks are always owned by a thread.
(iv) A Mutex variable initialized using PTHREAD_MUTEX_INITIALIZER can be used for process synchronization.

Index number: .....

Q19 What is false out of the following?

(i) Multiple semaphores initialed to zero can be used for thread serialization.
(ii) Only the owner thread of a mutex can unlock a mutex
(iii) Only the process or thread which called sem_wait( ) can call sem_post( ).
(iv) Using mutually exclusive locks in Simultaneous Multi-Threading (SMT) without initializing may lead to deadlocks.

Q20 What is false out of the following?

(i) Thread synchronization is not mandatory when each thread update its own local variable.
(ii) A concurrency control mechanism is required for hyper-threading applications that update the data segment variable of virtual address space.
(iii) Multiple threads updating a static variable defined inside a function do not cause inconsistent updates.
(iv) A semaphore can be used in shared memory to control simultaneous updates of the shared memory.

UNIVERSITY OF RUHUNA

Faculty of Engineering

End-Semester 6 Examination in Engineering: January 2022

Module Number: EE6206

Module Name: Operating Systems and Programming (N/C)
Part II (Essay)

[Two Hours and Twenty Minutes]

[Answer **all questions**, each question carries 10 marks. Mention all the necessary header Files in each of the program you write.]

---

1)

a) Write Linux console-based commands for the following sequence of operations?

(i) Go to home directory. Change the directory to **Desktop**. Display the current working directory.

(ii) Assuming there is a text file known as **txt1.text**; create an exact copy of **txt1.text** as **txt2.text** in the previous (home) directory.

(iii) Go back to the **home** directory and copy the **Desktop** directory to inside the **Documents** directory as **Desktop_copy**. Note that **Documents** is a directory inside the home directory.

(iv) Delete **Desktop_copy**, go to the **Home** directory and list the details of **txt2.text** file in long format.

(v) Remove the read privilege of the user for **txt2.text** and add write privilege to users in File's group.

(vi) Create a directory named **mydir2** in **Desktop**. Go to that directory. After that create two empty text files as **Doc1.txt, Doc2.txt** inside **mydir2** using,

(I) **Touch** command

(II) **Cat** command

(vii) Assuming both **Doc1.txt** and **Doc2.txt** has content inside them, copy the content of **Doc1.txt** to **Doc2.txt** and display **Doc2.txt**.

(viii) Observe the statistics of **Doc1.txt.** Change the access time of **Doc1.txt.** Change the modification time of **Doc1.txt**

(ix) Assume there is a C program file called my_program.C. Write instructions to compile and run an object file with same name as my_program.C

[4 marks]

b) Write a code using C programming language for the following.
Create a **line** structure by nesting two (x, y) **point** structures. Get input from user for the 2 points and calculate and display its length using the knowledge of pointers to structures and functions.

[3 marks]

c) Write a code using C programming language for the following.
Using the High-level I/O model only, create and open a file **doc1.txt** with read and write privileges. Then, write text *"Hello, this is OS programming."* to **doc1.txt**. Then, append the content of **doc1.txt** to the end of a file named **doc2.txt**. After that, change the privilege of **doc1.txt** to write only. At last, the program must try to read from **doc1.txt** with error handling.

[3 marks]

(Q2)

a)

(i) State two (2) differences and two (2) similarities between structures and unions.

[1 mark]

(ii) Briefly explain the meaning of **'pass by reference'** for functions using a simple example.

[1 mark]

b)

(i) State two (2) differences and two (2) similarities between pipes and message queues.

[1 mark]

(ii) State and briefly explain different ways to create unique keys for generating unique Inter Process Communication (IPC) objects.

[1 mark]

c)

(i) Write a complete C program to implement the following scenario.

Use Inter process communication using pipes to implement the following scenario for 3 processes **P1, C1, C2** where **P1** is the initial process, **C1, C2** are the children of P1

The root parent **P1** must send the message **"Hello my child"** to each of its children and waits for response. When any child process **(C)** receives the message **"Hello my child"** from its parent process **(P)** and if that received child process **(C)** does not have any children it must immediately respond to the parent process **(P)** as **"Hello mum"** and terminate its process. If that received child process **(C)** has children; then it must send **"Hello my child"** to each of its children and wait until each of the children respond with **"Hello mum"**. After each of children has responded with **"Hello mum"** to **C**; then only **C** must respond to its parent **(P)** as **"Hello mum"** and terminate its process **(C)**. When each process receives a message; it should print a statement displaying its name (C1, P1 etc.), its process ID, the sender's process ID and the particular message received to standard output. The program must include error handling as well.

[3.0 marks]

(ii) Write a complete C program to implement the following scenario.

An initial process **(P1)** creates a child process **C1**. **C1** creates its child process **C2**. Create a message queue and send the following messages to the queue from the initial process **(P1)**. **P1** must send the messages to the queue in the exact order given in Table Q2 starting from **"MESSAGE 1"** and wait until its child process **(C1)** terminate.

TABLE Q2: Table of messages in the message queue

Type	Text
25	MESSAGE 1
50	MESSAGE 2
75	MESSAGE 3
50	MESSAGE 4
35	MESSAGE 5
110	MESSAGE 6

- Suspend the execution of C1 for 1 second after P1 creates the queue. After that 1s, C1 must read and print first 2 messages of the queue in FIFO manner and print to terminal along with the process ID. After that, C1 will wait until C2 terminates. Once C2 terminates, C1 will also terminate.
- Suspend the execution of C2 for 3 seconds after P1 creates the queue and then read next 2 messages in ascending order of type of messages. Then, C2 must terminate after printing the 2 messages it read.
- After the child process C1 terminates, the initial process (P1) must read the message with type 110 and it should terminate after printing the read message with process ID.

What is the message remaining in the queue at last?

[3.0 marks]

(Q3)

a)

(i) State the corresponding function used for sending a set of bytes from a memory buffer to a file in each of Low level I/O and High level I/O models. Mention two (2) differences and two (2) similarities between those two functions used in two types of models. You may discuss about the intended task, number and type of arguments, return types, header files etc. for comparison.

[1 mark]

(ii) List two (2) different types of permission identified in file systems.

[0.5 mark]

(iii) A programmer has passed **0666** as the third argument of *open( )* function. Write an alternative way to pass this argument to *open( )* function with the required header file.

[1 mark]

b)

(i) Define what is a thread and a process separately. Specify two (2) differences and two (2) similarities between threads and processes.

[1.5 mark]

(ii) State 3 ways that can be used for Inter process communication and 2 techniques of thread synchronization.

[1 mark]

c)

(i) Write a complete C program to implement the following scenario.

A parent process **P1** will create two child processes **(C1, C2)** simultaneously. Create a shared memory segment storing a long value initialized to zero. **C1** must run a function which increments the value of the shared memory segment by one (1) 10 million times whereas **C2** increments it by two (2) 10 million times. At the same time, the parent process **(P1)** must decrement it by three (3) 10 million times. At last, the parent process must return the value of shared memory segment. Use concurrency control using Semaphores to provide the correct result.

[2.5 marks]

(ii) Write a complete C program to implement the following scenario.
Create a hyper-threaded process with two (2) simultaneous threads. One thread should write as "**T1**" repeatedly forty (40) times and the other thread should write as "**T2**" repeatedly forty (40) times. Thread synchronization should be as follows. When **Thread1** executes, it should print the word "**T1**" ten (10) times and after that wait until the other thread prints "**T2**" five (5) times before **Thread1** prints again. After **Thread1** has finished printing; **Thread2** can print continuously. Use Mutex and condition variables for this program.

[2.5 marks]

Q4)

a)

(i) Define what is a socket in IPC and describe the *connect( )* function.

[1 mark]

(ii) List down the 7 attributes of a socket.

[1 mark]

(iii) State and briefly explain the two types of byte orders.

[0.5 mark]

(iv) Briefly explain the meaning of the following two commands,

(I) cat /etc/hosts

(II) netstat -ant | grep 22

[1 mark]

(v) State and briefly explain the server only functions in the correct sequence which they should be called in a server machine in client-server communication.

[1.5 mark]

b)

(i) Write a complete C program to implement the following scenario.

Create a client program to connect to the **echo** service of a remote machine to send a user specified string of user specified size and display the returned string on screen. Include error handling for sending only. The dotted decimal notation of the IPv4 address of the remote machine is **148.158.56.521**. The socket's domain is internet and its service type is stream. After receiving the reply from the remote machine, the client program must make sure that the socket is disabled for reading only for any process which has been connected to it.

[2.5 marks]

(ii) Write a complete C program to implement the following scenario.
Create a "simultaneous repetitive calculation" program as follows using C programming language. Use 4 hyper threads (**T1, T2, T3, T4**) each returning values for Addition, Subtraction, Multiplication, Division respectively of user input two floating point numbers to the main thread when each of them terminates causing the main thread to print each returned value along with thread ID. Each thread T1, T2, T3, T4 must respectively print strings "**Addition** ", "**Subtraction** ", "**Multiplication**", "**Division**" concurrently user specified number of times. Use one thread function for all 4 threads where each thread passes different arguments to the thread function. There are no concurrency control measures employed. Assume each thread consumes a constant time of 1 ms to print its string "Addition", "Subtraction" etc.

[2.5 marks]