1) IS It a good Process to have declarations in between of C-Program)

A: For Compilers greater than C89 standard, it doesn't matter. All declarations are compiled at one place during Compilation Process. However), if you want to maintain Backwards Compatibility, declaring all variables at the begenning of the scope would be ideal.

2) Which syscall does fopen() call?

A: In Linux/Unix fopen(); Calls the open(); function

3) Example of code in C with open(); systemcall.

A:
```
#include <unistd.h>
#include <fcntl.h>
int main() {
        int fd = open ("text.txt", O_WRONLY | O_APPEND);
        /* open file */
    write( fd, "Data to be written\n", 20);
        /* write to file */
    return 0;
  }
```

4) Does the Code in ISR have Previliged instructions? if so give an example. If no, justify.

A) Yes, ISRs can have previliged instructions

Example! In the Case of an I/O request / I/O ready event, the ISR associated with it must interact withe the I/O device Controller directly.

5) Choose 2 Shells and explain the difference in executing similar Commands.

Bash vs ~~Korn~~ C-shells

~~Bash~~	Bash	C-shell
Default Prompt	$	%
force redirection	>|	>!
Home dir	$HOME	$home
variable assignment	var=value	set var=value
environmental variables	export var=value	setenv var value
Number of Arguments	$#	$# argv
aliases	alias x='y'	alias x y

Conditionals	if [$ -eq 5]
	if
		if ($i == 5)
		endif.

6) Write a Short note on POSIX.

POSIX → Portable Operating System Interface ⇒

→ Family of standards specified by IEEE for maintaining Compatibility between OSes.

→ Defines the API, CLI shells, interfaces, for compatibility with UNIX & other OSes

→ Consists of 19 documents → (Posix.1, Posix.2 etc)

→ CLI & scripting based on UNIX-V

→ Also defines a standard threading API

7) When installing OS, why is choice of 32-bit or 64-bit given, but not of the Processor)

A: 32- or 64 bits describe the architecture of Processor. a 64 bit system can reference more memory than a 32-bit system. A 64 bit System can run 32-bit Software but not vice versa.

Hence, while installing, you get the choice of ~~the~~ either 32, or 64 bit installation, as 64 bit Processors can also run 32 bit OSes, but the Processor inside is fixed. Hence, we don't get a choice of Processors.

8) Why are buffers Passed in syscalls?

A: Buffers are used in syscalls, either to read from or to write to, them.

For example, in read(), the data from file is read and placed into the buffer, from where we can access it

in write(), the data to be written is stored in buffer.

9) Why is Stack Pointer incremented when Library Call returns to user.

A: -> Stack Pointer ~~is~~ movement depends on direction of growth of the stack.

-> If stack is an upward growing one, then, ~~f~~when tasks are pushed, Pointer increases & when tasks finish, Pointer is decremented

-> However in downward growing stacks, stack Pointer increments when tasks finish / return

-> Hence if stack Pointer increases after returning from library call, it must be a downward growing one.

10) What happens in a read(); call?

A: → read → ssize_t read(int fd, void *buf, size_t count)

→ reads atmost "count" number of bytes from the specified file descriptor (fd) into the buffer (buf)

→ fd → file descriptor. read starts reading the data at fd, and keeps copying it to the buffer.

→ once "count" number of bytes have been read, process terminates

11) Give 5 examples for preemptive scheduling. Identify state of process in each

A) 1) Shortest Remaining Time scheduling
→ Process with shorter remaining time preempts the one with larger time
→ Running process may be preempted by a new process with shorter ~~run~~ time.

2) Round Robin scheduling.
→ Each process is switched after a quantum 'q'
→ Here, ~~all~~ Running process is preempted by other process in the queue

3) Multilevel Queue scheduling
→ Multiple queues: each queue follows own algorithm.
→ Process preempted and moved to lower queue if it
ⓐ takes longer than quantum
ⓑ performs I/O operation.

4) Rate Monotonic scheduling:
→ process with smaller period preempts the other process
→ Running process is preempted if new process is initiated with a shorter period

5) Earliest Deadline First scheduling
- → Process with earlier deadline preempts other process.
- → Running process is preempted if new process arrives with an earlier deadline

12) Given deadline & Period, until what point should we draw the schedule so as to ensure no deadline misses occurs.

A: → Deadlines are periodic functions of time, i.e. They repeat with a certain time period

→ The same applies to the pattern of the processes too.

→ If there are 'n' processes with their own ~~unique~~ periods, the period of resultant system would be the LCM of all periods

→ If there is no deadline miss in one period of the whole process system, we can say that there would be no deadline misses in any subsequent periods (unless new process is added)

→ Hence LCM of all periods gives us the point until which we should draw. the schedule.