1	2	3	4	5	6	7	Total

GIA IA-UAB june 14th 2023

Computer Fundamentals

Surname:

Name: DNI or NIU:

1.- [2 p.] Write a C program where a father process creates 3 concurrent sons and then waits for the sons to end. Use C process creation system calls and a for or while loop for the creation of the processes.

```
for(i=0; i<3; i++){
   id1=fork();
   if(id1==0){
      printf("I am a son process: %d", i);
      exit();
   }
}
for(i=0; i<3; i++){
   wait();
   printf("%d son process finished",i+1);
}
printf("all son processes have finished\n");
```

2. -[2 p.] Is this code functionally correct? Why? What changes should you add to solve the problem?

```
int counter = 300;

void* pthread_function(int id) {
  int counter_read = counter;
  counter = counter_read + 1;
  pthread_exit(NULL);
}

int main(int argc,char** argv) {
   pthread_t thread[2];

   pthread_create(thread[0],NULL,(void*)pthread_function,(void*)(0));
   pthread_create(thread[1],NULL,(void*)pthread_function,(void*)(1));

   pthread_join(thread[0],NULL);
   pthread_join(thread[1],NULL);

  printf("[Master thread] Finished\n");
  return 0;
}
```

The code has a critical section problem when different threads want to modify the value of a shared global variable counter. Write access to this variable should be protected with a synchronization mechanism as a mutex. We should define a mutex and a use its synchronization primitives to manage the shared access to counter_read variable:

```
void* pthread_function(int id) {
  lock(m);
  int counter_read = counter;
  counter = counter_read + 1;
  unlock(m);
  pthread_exit(NULL);
}

int main(int argc,char** argv) {
   pthread_t thread[2];
   mutex m;
```

3. -[1 p.] What is a container? Which are the most relevant advantages that containers provide?

A container is a standardized unit of software designed to run quickly and reliably on any computing environment that runs the containerization platform.

Containers provide operating system (OS) virtualization so that you can run an application and its dependencies in resource-isolated processes. A container is a lightweight, standalone software package that contains everything that a software application needs to run. For example, it can contain the application code, runtime engine, system tools, system libraries, and settings. Containers can help ensure that applications deploy quickly, reliably, and consistently regardless of the deployment environment.

4. -[1 p.] What is a critical section? Write an example of a critical section and which solution could be applied to give a solution to the problem in the example.

If we have some threads **accessing writable shared** data, access to this data must be protected. A Critical section is a number of code instructions that access a shared resource that must be executed by one thread only at each instant.

A synchronization mechanism for threads is a mutex;
Two atomic operations:
lock(): thread tries to lock mutex. If mutex is locked, thread blocks
unlock(): thread unlocks mutex and wakes up a blocked thread

```
int cont=0;
main()
{
  mutex m;
  create_thread(t0);
  create_thread(t0);
}

void t0()
{
  lock(m);
  cont++;
  unlock(m);
}
```

5. -[1 p.] What is a Dockerfile? Which is the effect of the lines of the Dockerfile in the system? Which are the main benefits of Docker containers?

Docker containers are created from read-only templates, which are called container images. Images are immutable and highly portable. You can port an image to any environment that supports Docker. Images are built from a Dockerfile, which is a plain text file that specifies all of the components that are included in the container.

Instructions in the Dockerfile create layers in the container image. Dockerfiles are Step-by-step instructions to create container images. The operations defined in dockerfile are done inside the "virtual" image that is being built. To execute a command, we can use RUN operation like RUN echo "hello world".

The following list summarizes some of the important benefits of Docker containers:

- Docker is a portable runtime application environment.
- You can package an application and its dependencies into a single, immutable artifact that is called an image.
- After you create a container image, it can go anywhere that Docker is supported.
- You can run different application versions with different dependencies simultaneously.

These benefits lead to much faster development and deployment cycles, and better resource utilization and efficiency. All of these abilities are related to agility.

6. -[1 p.] What is multiprogramming? Can a system with one CPU execute more than one process during one second? why?

Multiprogramming is the ability to run multiple programs at the same time.
When a program waits for I/O operations, SO executes other processes, so if we use the idle intervals of a process, other processes could use the CPU so that some process would share the same CPU.

7. - [1 p.] Which are the main components of a process? What is the PCB and which information is stored?

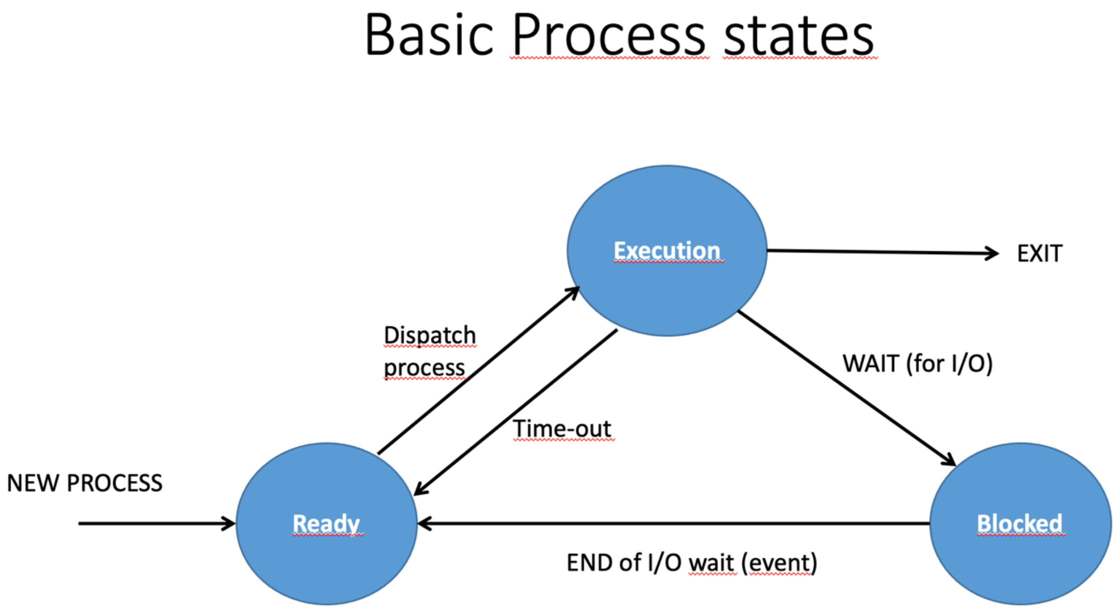
A process is made of:

- Program code: instructions
- Program data: variables
- Data associated with program execution
- CPU registers
  - State
  - Input/output information
  - Priority

Process Control Block (PCB) is the OS table to keep processes management information.

- CPU state (registers)
- Identification information: process id, user id,
- Control information: process state, memory assigned, open files

8. - [1 p.] Which are the main states of a process? Draw the process states of the life of a process and their relationships in a graph


Basic Process states
Execution
EXIT
Dispatch process
WAIT (for I/O)
Time-out
NEW PROCESS
Ready
Blocked
END of I/O wait (event)