# OBSERVACIONES DEL LA PRACTICA

Laura Valentina Guiza Cod 201920926
Alejandro Cruz Cod 201912149.
Sebastián Montoya Alvarez Cod 202317398.

## Preguntas de análisis

1. ¿Por qué en la función getTime() se utiliza time.perf_counter() en vez de otras funciones como time.process_time()?

   time.perf_counter() se utiliza porque en esta función se puede tomar el tiempo de una manera no especifica y es más apta para medir el tiempo de manera real que time.process_time(). La cual solo funciona de manera más específica y no tiene en cuenta el tiempo de espera mientras el programa este suspendido.

2. ¿Por qué son importantes las funciones start() y stop() de la librería tracemalloc?

   Son importantes porque estas start() da el inicio del contador mientras que el programa se esté ejecutando y a su vez toma el número de fotogramas que se van almacenado. Mientras stop() se encarga de borrar los residuos que deja la función start() y evita que se quede archivos cache.

3. ¿Por qué no se puede medir paralelamente el uso de memoria y el tiempo de ejecución de las operaciones?

   Porque estos procesos si actuaran al mismo tiempo podrían consumir muchos recursos del equipo generando una alta demanda en la memoria ram y causando posibles problemas en el programa y equipo.

4. Teniendo en cuenta cada uno de los requerimientos del reto ¿Cuántos índices implementaría en el Reto? y ¿Por qué?
   2 , porque uno es para partidos y otro para jugadores.

5. Según los índices propuestos ¿en qué caso usaría **Linear Probing** o **Separate Chaining** en estos índices? y ¿Por qué?

   Según los índices propuestos, se usaría Linear Probing en los índices que tienen una alta probabilidad de colisiones, ya que es más eficiente en términos de memoria y tiempo de ejecución. En cambio, se usaría Separate Chaining en los índices que tienen una baja probabilidad de colisiones ya que es más eficiente en términos de memoria y tiempo de ejecución. Por ejemplo, para el índice de jugadores, se usaría Separate Chaining debido a que la cantidad de elementos es relativamente baja y no se espera una alta probabilidad de colisiones.

6. Dado el número de elementos de los archivos del reto (large), ¿Cuál sería el factor de carga para estos índices según su mecanismo de colisión?
   0.5, se halla dividiendo la cantidad de elementos entre la capacidad de la tabla hash. En el caso de Linear Probing, la capacidad de la tabla hash se define como el doble de la cantidad de elementos,

mientras que en el caso de Separate Chaining, la capacidad se define como la cantidad de elementos dividida entre el factor de carga.

7. ¿Qué cambios percibe en el **tiempo de ejecución** al modificar el factor de carga máximo para cargar el catálogo de contenido FIFA?

   Si se aumenta el factor de carga, el tiempo de ejecución disminuye, ya que se reduce la cantidad de elementos que se deben buscar en la tabla hash. Pero, si se disminuye el factor de carga, el tiempo de ejecución aumenta, ya que se aumenta la cantidad de elementos que se deben buscar en la tabla hash.

8. ¿Qué cambios percibe en el **consumo de memoria** al modificar el factor de carga máximo para cargar el catálogo de contenido FIFA?

   Si se aumenta el factor de carga, el consumo de memoria también aumenta, ya que se deben almacenar más elementos en la tabla hash. Por otro lado, si se disminuye el factor de carga, el consumo de memoria disminuye, ya que se almacenan menos elementos en la tabla hash.

9. ¿Qué cambios percibe en el **tiempo de ejecución** al modificar el esquema de colisiones?, si los percibe, describa las diferencias y argumente su respuesta.

   Si se cambia de Linear Probing a Separate Chaining, el tiempo de ejecución puede disminuir en los índices que tienen una alta probabilidad de colisiones y si se cambia de Separate Chaining a Linear Probing, el tiempo de ejecución puede disminuir en los índices que tienen una baja probabilidad de colisiones.

10. ¿Qué cambios percibe en el **consumo de memoria** al modificar el esquema de colisiones?, si los percibe, describa las diferencias y argumente su respuesta.

    Si se cambia de Linear Probing a Separate Chaining, el consumo de memoria puede aumentar en los índices que tienen una alta probabilidad de colisiones, ya que Separate Chaining requiere más memoria para almacenar los elementos. Y, si se cambia de Separate Chaining a Linear Probing, el consumo de memoria puede disminuir en los índices que tienen una baja probabilidad de colisiones.

11. ¿Qué configuración de ideal ADT Map escogería para el **índice jugadores FIFA (“scorer”)** ?, especifique el mecanismo de colisión, el factor de carga y el numero inicial de elementos.
    Se escogería un ADT Map con Separate Chaining como mecanismo de colisión, un factor de carga de 0.5 y un número inicial de elementos de 1000. Esto se debe a que la cantidad de elementos es relativamente baja y no se espera una alta probabilidad de colisiones, por lo que Separate Chaining es la técnica más eficiente en términos de memoria y tiempo de ejecución en este caso. Además, un factor de carga de 0.5 permite un buen equilibrio entre el tiempo de ejecución y el consumo de memoria. El número inicial de elementos se escogería en función de la cantidad de elementos que se espera almacenar en el índice.