Antes de empezar el examen, leed atentamente los siguientes comentarios:

- Esto es un examen y los exámenes se hacen de forma **individual y sin ayudas externas**.
- La fecha límite para entregar el examen (y el proyecto) es el **miércoles 17 de enero de 2024 a las 12:00**.
- Hemos montado una "práctica" en el racó para hacer la entrega del examen.
- El examen que entreguéis ha de ser un **PDF**.
- Hemos calculado que el tiempo necesario para realizar el examen son unas 20-30 horas de trabajo. No lo dejéis para el final.
- Teniendo en cuenta que tenéis mucho tiempo para hacer el examen, una buena redacción y ortografía se tendrá en cuenta en la evaluación.
- No os limitéis a la información que hay en las transparencias del curso, hay que buscar en otras fuentes.
- Además del examen tenéis que entregar un anexo, en el que, para cada pregunta, indiquéis las referencias, páginas web, etc. que habéis consultado durante el examen.
- El examen consta de:
  - ✓ 6 preguntas estándar (2 hojas por pregunta, aprox. 1000 palabras por pregunta).
  - ✓ 1 pregunta doble, es cómo un pequeño trabajo (4 hojas, aprox. 2000 palabras)
  - ✓ 1 cuestionario, que equivale a 1 pregunta doble.
  - ✓ Los que hicisteis la actividad práctica el 27 de marzo (servidor de TGA) podéis dejar de hacer 1 de las preguntas estándar (de la 2 a la 6, la 1 es obligatoria).

- No os olvidéis de poner vuestro nombre al inicio del examen.

## PREGUNTAS ESTÁNDAR

1) Describe el pipeline gráfico tradicional.

2) Dada la siguiente rutina escrita en C:

```
void Exa24(float mC[N][M], float mB[N][M], float vA[N]) {

  for (int i=0; i<N; i++)
    for (int j=0; j<M; j++)
         mC[i][j] += mB[i][j]*vA[i] + mB[0][j]*mB[i][0];
}
```

Escribid 3 versiones de un kernel CUDA que resuelva el mismo problema:

a) En la primera versión cada thread se va a ocupar de 1 columna de la matriz resultado.
b) En la segunda versión cada thread se va a ocupar de 1 fila de la matriz resultado.
c) En la última versión cada thread se va a ocupar de 1 elemento de la matriz resultado.

Escribid los kernels CUDA para cada versión, así como la invocación correspondiente. Tened en cuenta que como máximo podéis utilizar 1024 threads por bloque y que las variables N y M pueden tener cualquier valor (p.e. N = 1237, M = 2311, suponed que N, M > 1024).
No es necesario que escribáis las transferencias de información (CPU→GPU, GPU→CPU).

3) ¿Cuál crees que es el fabricante de tarjetas gráficas más importante? Justifica la respuesta. A la vista de cómo han ido evolucionando las tarjetas. Selecciona la tarjeta que consideres más relevante de la historia. Justifica porqué has elegido esa tarjeta y describe sus características más relevantes. Es más importante la justificación que la tarjeta escogida.

4) Hablando de gráficos 3D, ¿qué es el aliasing?, ¿por qué se produce?, ¿qué hacen las tarjetas gráficas para evitarlo?, ¿la IA, es una buena alternativa para resolver este problema

5) Si pensamos en los parámetros que definen una tarjeta gráfica: número de cores, ancho de banda, cantidad de memoria, tipo de memoria, familia, … Enumera desde tu punto de vista cuales son los 10 más importantes (ordenados de mayor a menor importancia) y haz una descripción de cada uno de ellos.

6) Disponemos de una tarjeta gráfica con 2 GPUs. En esta tarjeta queremos correr un juego interactivo 3D (que utiliza OpenGL u otra API similar). Si estuvierais diseñando el driver de la API gráfica, ¿cómo distribuirías el trabajo entre las 2 GPUs para maximizar el rendimiento? ¿Qué información hay que enviar a cada tarjeta? ¿Han de sincronizarse/comunicarse las 2 GPUs? ¿Cómo pueden hacerlo? Os ayudará tener en mente cómo funciona el pipeline gráfico tradicional.

## PREGUNTA DOBLE

Hablando de las tarjetas gráficas de NVIDIA. Explica qué son los Tensor Cores, cómo funcionan, cómo se programan, para qué sirven, y las diferentes variantes que existen. Os ayudará que esta pregunta la encaréis cómo si fuera un pequeño trabajo.

## CUESTIONARIO

Las 5 últimas cuestiones (f-j) se contestan con 1 frase, 1 punto por pregunta. Las 5 primeras (a-e) requieren una explicación más elaborada, 3 puntos por pregunta.

a) Siempre decimos que en una GPU ocultamos la latencia con memoria. ¿Puedes explicar cómo lo hacemos?

b) ¿Qué significa que una GPU tenga los shaders unificados?

c) Hablando gráficos 3D, ¿qué diferencias hay entre CLIPPING y CULLING?

d) En la declaración de un kernel CUDA, proponemos dos alternativas:

```
__global__ void Kern(int N, float *A, float *B);
__global__ void Kern(int N, float *restricted A, float *restricted B);
```

¿Qué implicaciones tiene el uso de `restricted`? O mejor, ¿qué implica NO usarla?

e) Hablando de gráficos 3D, ¿qué es una textura? ¿para qué se utilizan?

f) ¿Cuál es la tarjeta gráfica más potente hoy (12/dic/2023 – 17/ene/2024)? Indica la fecha de la consulta.

g) ¿Cuál es la mejor tarjeta gráfica que te puedes comprar hoy (12/dic/2023 – 17/ene/2024) por 700 euros? Indica la fecha de la consulta y dónde la puedes comprar.

h) ¿Cómo hay que declarar un vector de 256 floats dentro de un kernel de CUDA para que se almacene en la memoria compartida de la GPU?

i) Enumera los elementos programables (shaders) que hay en el pipeline de direct3D 12.

j) Pensando, en una tarjeta gráfica, explica que es el FILL RATE.