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

- Esto es un examen y los exámenes se hacen de forma **individual**.
- La fecha límite para entregar el examen (y el proyecto) es el **lunes 21 de junio de 2021 a las 8:00**.
- He 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.

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

PREGUNTAS

1) Describe el pipeline gráfico tradicional.

2) Dada la siguiente rutina escrita en C:

```
void Examen21(float mA[N][M], float mB[N][M], float vC[N], float vD[M])
{
  int i, j;
  for (i=0; i<N; i++)
    for (j=0; j<M; j++)
      mA[i][j] = mA[i][j]*vC[i] - mB[i][j]*vD[j] + mA[i][0]*mB[7][j];
}
```

Escribid 3 versiones del 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).

3) 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.

4) Una de las herramientas que utilizamos en CUDA son los eventos (**`event`** en CUDA). ¿Para qué sirven? ¿Cómo se utilizan? Pon ejemplos de uso.

5) Si queremos utilizar GPUs para cálculo de propósito general (GPGPU) puedes escoger entre CUDA, OpenCL y OpenACC. Describe las ventajas e inconvenientes de cada alternativa. Además, se pueden combinar. ¿Qué posibilidades ofrece combinar OpenACC con CUDA o OpenCL?

6) Hablando de texturas, ¿qué filtros existen?, ¿puedes describirlos? ¿qué implicaciones tienen en el diseño de la GPU?

7) Pregunta doble. Explica qué es el raytracing, cómo funciona, y cómo se implementa en las nuevas tarjetas de Nvidia, las RTX. Os ayudará que esta pregunta la encaréis cómo si fuera un pequeño trabajo.

8) Cuestionario. Pregunta Doble. Las 5 últimas cuestiones (f-j) se contestan con 1 frase. Las 5 primeras (a-e) requieren una explicación más elaborada.

   a) ¿Cual es la diferencia entre un fragmento y un píxel?
   b) ¿Qué hace el rasterizador?
   c) Siempre decimos que en una GPU ocultamos la latencia con memoria. ¿Puedes explicar cómo lo hacemos?
   d) ¿Qué significa que una GPU tenga los shaders unificados?
   e) ¿Para qué sirve el Z buffer? ¿Cómo lo hace?
   f) Define en una frase CLIPPING
   g) Define en una frase CULLING
   h) ¿Qué es REYES?
   i) ¿Qué es el Aspect Ratio?
   j) Muchas veces hablamos de GPGPU, ¿que quieren decir estas siglas?