Work Test Assignment

When taking a work test, it's important not only to show mastery of your given discipline (in my case, systems and gameplay programming), but also to show that you have an intimate understanding of the products that the company makes, their tools, and their processes. For most companies, finding what game engine they use is trivial. However, due to the secretive nature of our industry, there are a lot of times where you'll need to read between the lines on what sort of tech stack or frameworks they'll likely be employing. For example, if they are a studio that has previously made only large scale multiplayer games, making sure that you tailor your work sample to be compatible with that type of networking is key.

Making a relevant work sample isn't just about demonstrating your understanding of what the company does, though. Showing that you're familiar with the day-to-day tasks and needs of the company is also a huge component. Making sure to pick a project that is of comparable scope and feature set to what the actual employees of the company are making is necessary. Not only does this show that you are a competent worker who knows what it's like in the industry, it'll also help create a more clear path towards planning what you're going to actually make.

Some work sample requests will ask for specificity, and already lay out the plan for what you're making. An assignment like "Animate a cartoonish walk cycle of a happy-go-lucky character. Pick a model off Mixamo for the base, or use an existing character you've made.", is extremely straightforward. However, sometimes you'll be faced with an assignment that gives you a lot of freedom, and you'll be tasked with putting on your designer hat to decide what to create. You have to make sure to pick systems that speak to your unique capabilities as a potential employee, while also using the methods listed above to make sure it's relevant to the company. If you're having trouble coming up with potential ideas, try mixing and matching different facets of games the company makes with some from their competitors. If you're still stuck and keep coming up short, turning to tools like ChatGPT to kickstart the brainstorming process can be helpful for some people. Just make sure that you're adding your own original twist, and taking it beyond just the initial concept.

The work test I was recently given was provided by TTK Games, a studio founded by former Dice employees. I was given an Unreal project consisting of a tank that can move around and shoot static enemies. However, it had several bugs, and would sometimes crash. My task was to first get rid of the bugs and the crashing issues, implement a health system into the game, and then to implement something that I was passionate about. Some examples that they gave were things like homing missiles, more intricate tank movement, or bullet pooling.

The project's bugs were a lot of smaller things here and there, which made finding and fixing them quick. For example, a function that is supposed to release a reference of a bullet in a subsystem instead tried adding itself again, causing a null pointer error. Another bug was a piece of code that was executed twice in two colliding objects, where both objects tried deleting each other and then themselves.

After fixing all the bugs and crashes, I wrote a health component class for the enemies. I made sure to keep it as generic as possible, so that it could be reused for anything else that might have health, like the player or breakable objects.

For the final task, adding a new feature I'm passionate about, I chose to add a weapon system. The original gun for the tank that the project came with was extremely simple, lacking the sophistication that most games have. All it did was spawn a mortar-style arced projectile whenever the player clicked. Since the tank's movement was sufficient in my opinion, I felt that adding a more elaborate system for shooting had the most opportunity for demonstrating my skills.

The system includes a weapon handler component that is attached to whatever needs to use weapons. It holds references to the weapons the user currently has available, and includes functions for switching between weapons. Weapons themselves have values for how much max ammo can be held, how much ammo fits in its magazine, how long it takes to reload, how long you must wait between shots before you can shoot again, what type of projectile it should spawn, and more.

As examples to show the versatility of the weapon system, I included three different unique weapons on the tank that can be switched between. The first is taken directly from the original

project, but the other two were made by me. One of them fires rapid projectiles wherever the player is aiming, like a machine gun. The other functions like homing missiles. I chose the homing missile specifically because it was listed as an example of something to add to the game in the original work sample brief, to show that not only could I accomplish the example listed there, I could create an entire scalable system to support a wide variety of weapons as well. This task took me a day to complete. Fortunately, the team at TTK Games said that they were impressed with what I'd delivered, and I'm now on track towards the next round of interviewing.