# CS1112 Programación II

**Proyecto Final**
**Pregrado**
**2023-2**
**Profesor: Ian Paul Brossard**
**Lab 2.03**

---

**Proyecto Final [BWAPI] - Análisis de Eficiencia de Recursos en StarCraft**

**Descripción General del Proyecto**

En este proyecto final, su objetivo es desarrollar un bot en C++ que mida la eficiencia de los campos de minerales y fuentes de gas en el juego StarCraft 1.16, utilizando la Brood War API (BWAPI) y partiendo del bot STARTcraft de Dave Churchill. Este bot no participará en combate sino que se centrará exclusivamente en la recolección de datos para analizar la eficiencia de recolección de recursos.

**Instrucciones Detalladas**

Preparativos Iniciales

1. **Instalación de Visual Studio**:
    - Descargue e instale Visual Studio 2022 Community desde el siguiente enlace: https://visualstudio.microsoft.com/es/downloads/
    - En el tipo de instalación, elige "Desarrollo para el escritorio con C++".
    - Marca las casillas de las versiones de MSVC v142, v141, y v140 para garantizar la compatibilidad con BWAPI.**Clonación del Repositorio STARTcraft**:
    - Utilice la funcionalidad de clonación de Visual Studio para obtener el repositorio de STARTcraft en su entorno local de desarrollo. El repositorio a clonar se encuentra en el siguiente enlace: https://github.com/davechurchill/STARTcraft

## Pregunta 1: Conteo de Frames entre Aumentos de Minerales

**Objetivo**: Establecer un sistema de seguimiento que cuente la cantidad de frames que pasan entre cada incremento de 8 unidades de minerales en la cuenta del jugador.

**Detalles**:

- Deberá registrar la cantidad actual de minerales del jugador al final de cada frame.
- Si la cuenta ha aumentado en un múltiplo de 8 desde el último frame, se deberá reiniciar un contador de frames.
- Utilice la función **drawTextScreen** para mostrar los resultados acumulados en pantalla.
- Muestre los siguientes elementos:

    - La cantidad de frames que se están contando (este número irá aumentando en cada frame hasta llegar a un incremento de minerales).
    - Cuál fue el último conteo antes de reiniciar el contador.
    - Los últimos 20 conteos, separados por comas.
    - El promedio de los últimos 20 conteos. En caso de tener menos de 20 conteos, deberá mostrar el promedio de todos los conteos.

**Sugerencias**:

- Utilice variables para mantener la cuenta anterior y actual de minerales.

## Pregunta 2: Eficiencia de Campo de Minerales Individual

**Objetivo**: Implementar un contador individual para cada campo de minerales que mida cuántos frames pasan desde la última disminución de 8 unidades de dicho mineral.

**Detalles**:

- Monitoree la cantidad de minerales en cada campo de minerales en cada frame.
- Compare la cantidad actual de cada mineral, con la del frame anterior para determinar si se han recolectado minerales.
- En caso de que se recojan minerales, reinicie el contador de frames y utilice **drawTextMap** para mostrar la cantidad de frames transcurridos directamente sobre el campo de minerales correspondiente (para eso, usted deberá capturar la posición de cada mineral y usar dicha posición para la ubicación del TextMap).
- Muestre los siguientes elementos:
    - Cuál fue el último conteo antes de reiniciar el contador.
    - El promedio de los últimos 20 conteos. En caso de tener menos de 20 conteos, deberá mostrar el promedio de todos los conteos.

**Sugerencias**:

- Cree una estructura o clase que mantenga el estado de cada campo de minerales.
- Use la posición de cada mineral field para elegir una ubicación adecuada para sus textMaps (considere que si coloca el textMap en la posición del mineral, dicho textMap aparecerá en el centro del mineral. Posiblemente quiera ponerlo 32 o 64 pixeles a la izquierda.
- Dado que va a imprimir el último conteo, seguido por el promedio, se le sugiere poner un salto de línea entre el contador y el promedio. De esa forma, sus textMaps no se superpondrán sobre los minerales de los costados.
- Redondee o trunque el promedio a solo 2 decimales. De esa forma, sus textMaps no se superpondrán sobre los minerales de los costados.

- Debe ser posible medir la eficiencia de cualquier mineral del juego, no solo la de los minerales iniciales que se encuentran cerca de su ubicación inicial.

## Pregunta 3: Evaluación de la Eficiencia de la Recolección de Gas

**Objetivo**: Medir y mostrar la eficiencia de la recolección de gas mediante el seguimiento de la cantidad de gas recolectado y el tiempo en frames que tardan tres trabajadores asignados en recolectar incrementos de 8 unidades.

**Detalles**:

- En caso el jugador construya una estructura de gas (Refinery, Extractor o Assimilator), identifíquela y mida su eficiencia en base a la cantidad de recursos que tiene en cada frame. Cuente los frames hasta que la Refinery, Extractor o Assimilator pierda 8 recursos, tal y como trabajó la pregunta 2.
- Monitoree la cantidad de gas recolectada de dicha refinery, extractor o assimilator y el tiempo que toma recolectar cada recorte de 8 recursos.
- Use **drawTextMap** para mostrar la eficiencia de recolección en la ubicación de la estructura de gas.
- Muestre los siguientes elementos:
    - Cuál fue el último conteo antes de reiniciar el contador.
    - El promedio de los últimos 20 conteos. En caso de tener menos de 20 conteos, deberá mostrar el promedio de todos los conteos.

**Sugerencias**:

- No se preocupe en ordenar al bot que construya una refinery, extractor o assimilator. Eso lo hará el jugador humano. Usted solo evalúa la eficiencia de todas las refinerías de gas del juego.

## Documentación y Entrega

El código fuente deberá estar adecuadamente comentado y acompañado de un informe que explique su enfoque para cada una de las preguntas, los problemas encontrados y cómo los resolvió.

**Evaluación**

El proyecto será evaluado por la precisión en la recolección de datos, la claridad del código, la eficiencia del algoritmo implementado, y la profundidad del análisis presentado en su informe. Además, deberá realizar una presentación oral.

Esperamos que este proyecto le brinde una experiencia realista de cómo el análisis de datos y la programación en C++ se aplican en el desarrollo de videojuegos y en la investigación de la inteligencia artificial en juegos de estrategia en tiempo real.

**Rúbrica:**

| Criterio | EXCELENTE | ADECUADO | MÍNIMO | INSUFICIENTE |
|---|---|---|---|---|
| **Desarrollo de software** | Diseña y elabora el software para lograr una solución adecuada al problema planteado. El software debe ser ordenado, claro y óptimo.<br>(10 p.) | Diseña y elabora el software para lograr una solución adecuada al problema planteado. El software funciona pero no es ordenado, claro y óptimo.<br>(6 p.) | Diseña el software para lograr una solución adecuada al problema planteado. El software no se concluyó adecuadamente.<br>(4 p.) | No logra el diseño ni la implementación correcta del software.<br>(2 p.) |
| **Presentación escrita** | El informe contiene las secciones de Antecedentes, Fundamento Teórico, Métodos y Desarrollo y Conclusiones. Estas últimas, adecuadamente formuladas.<br>(5 p.) | El informe contiene las secciones de Antecedentes, Fundamento Teórico, Métodos y Desarrollo, pero no pone énfasis en las conclusiones.<br>(3 p.) | El informe contiene menos de la mitad de las secciones estipuladas, incluyendo conclusiones.<br>(2 p.) | El informe contiene menos de la mitad de las secciones estipuladas, sin incluir conclusiones.<br>(0 p.) |
| **Presentación oral** | El alumno presenta el proyecto en forma adecuada y responde a las preguntas del profesor en forma lógica y coherente.<br>(5 p.) | El alumno presenta el proyecto en forma adecuada, pero no responde a todas las preguntas del profesor en forma lógica y coherente.<br>(3 p.) | El alumno no presenta el proyecto en forma adecuada, pero responde a las preguntas del profesor en forma lógica y coherente.<br>(2 p.) | El alumno no presenta el proyecto en forma adecuada ni responde a las preguntas del profesor en forma lógica y coherente. O no se presenta a la presentación oral.<br>(0 p.) |