1.How is C programming used in embedded systems, operating systems, and game development?

1. Embedded Systems

Use Case:

C is the **dominant language** in embedded systems due to its performance, low-level memory access, and minimal runtime overhead.

Examples:

- **Automotive Systems**: Engine control units (ECUs), anti-lock braking systems (ABS), and airbag systems.
- **Consumer Electronics**: Microwaves, washing machines, smart TVs.
- **Medical Devices**: Pacemakers, diagnostic tools, and patient monitoring systems.

Why C?

- Direct hardware manipulation.
- Small binary footprint.
- Predictable performance for real-time systems.

2. Operating Systems

Use Case:

Most modern operating systems are either written in or heavily rely on C due to its close-to-hardware capabilities.

Examples:

- **Unix/Linux**: Core kernel, system libraries, and many utilities are written in C.
- **Windows OS**: Large portions of the Windows kernel and low-level system utilities are developed in C.
- **MacOS and iOS Kernels**: Based on XNU, which is primarily written in C.

Why C?

- Efficient system-level memory and process management.
- Portability across hardware platforms.
- Fine control over CPU and memory.

3. Game Development

Use Case:

While modern game engines use C++, C is still critical in **performance-critical components**, libraries, and graphics drivers.

Examples:

- **Game Engines**: Core libraries like physics engines, rendering backends (e.g., SDL, OpenGL libraries) often use C.
- **Console Game Development**: Firmware and hardware abstraction layers for consoles like PlayStation and Nintendo Switch.
- **Retro/Indie Games**: Developers use C for simplicity and performance on low-resource systems.

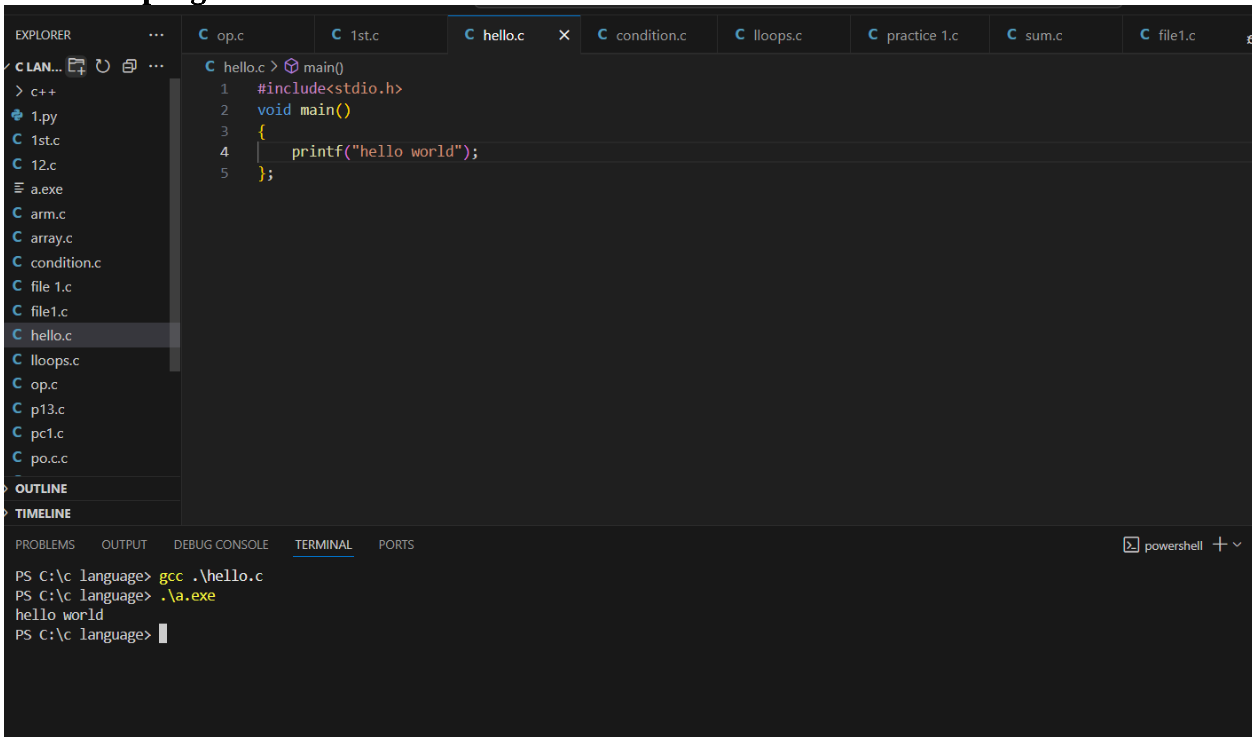
Why C?

- High performance and low overhead.
- Easier to port across platforms.
- Access to low-level graphics APIs like OpenGL and Vulkan.

2.Install a C compiler on your system and configure the IDE. Write your first program to print "Hello, World!" and run it.

1. **Install a C compiler:**
   - On Windows, download and install Code::Blocks (which includes MinGW compiler).
   - On Linux, install GCC using `sudo apt install build-essential`.
   - On Mac, install Xcode Command Line Tools with `xcode-select --install`.
2. **Configure the IDE:**
   - Open Code::Blocks or any IDE.
   - Create a new C project or file.

3. **Write the program:**



```c
#include<stdio.h>
void main()
{
    printf("hello world");
};
```

```
PS C:\c language> gcc .\hello.c
PS C:\c language> .\a.exe
hello world
PS C:\c language>
```

4. **run:**
   - In the IDE, click "Build" or "Compile".
   - Then click "Run" to see the output.

Output:

```
Hello, World!
```