**WEEK 9**

**KU ID: 100060665**

**NAME: Noura Alshamsi**

|  |
| --- |
| Q1.  Please write your GitHub Repository link for your week 9 Rustlings submission: |
| ANSWER:  <https://github.com/101failures/Lab7> |
| Q2.   1. Find and explain **two traits used in C2RUST**? (Please also **write the URL**s where you find those **trait**s!) 2. Explain where those are used, and why? 3. Explain the alternatives to using traits? |
| ANSWER:  A)  1-EventKindExt Trait: <https://github.com/immunant/c2rust/blob/3e0183e4d56f9d4d003f1ea2e9a814648debf307/pdg/src/builder.rs#L36>  It defines methods for handling events in the translation process.  It has two methods:   1. ptr(&self, metadata: &EventMetadata) -> Option<Pointer>;   Retrieves a pointer from an event using the provided metadata.   1. to\_node\_kind(&self, func: FuncId, metadata: &Metadata, address\_taken: &mut AddressTaken) -> Option<NodeKind>;   Converts an event into a NodeKind, likely for intermediate representation (IR) processing in translation.  2- AsStr Trait:  <https://github.com/immunant/c2rust/blob/3e0183e4d56f9d4d003f1ea2e9a814648debf307/analysis/runtime/src/parse.rs#L9>  It provides a method for returning a static string representation of an object.  It has one method: as\_str(&self) -> &'static str;  Returns a predefined string for a type.  B)  1-EventKindExt is used in event processing: it ensures a consistent way to extract pointers and convert events into node kinds.  2-AsStr is used for converting types into strings: it standardizes how different types return their string representations.  C)   * Without traits, each type would need its own method to achieve the same functionality. * Instead of AsStr, every type would require a separate fn to\_string(&self) -> &str method. * Instead of EventKindExt, event-related structs would need duplicated implementations for extracting pointers and converting to node kinds. |
|  |