Presidential Initiative for Artificial Intelligence and Computing (PIAIC)

PIAIC Batch 4-35 IoT

https://www.piaic.org

Internet of Things (IoT) Specialist Program

Quarter 2: Rust Programming

Assignment # 3

1. Run The following code and Explain the output in your wordings:
   a.
```
fn main(){
    Let r ;
    {
        Let x = 5;
        r=&x;

    }

    println!("{}",r);

}
```

ANSWER:
It'll give an error, because r is borrowing the x value and the scope of x is dropped before r is used.

1. Write a rust program:
   a. Define a struct Students with attributes (name , age , education, timing) all attributes must be reference of type str
   b. Impl a get_name method which Returns Student name as reference type
   c. Impl a get_timing method which returns Student timing as reference type
   d. Impl a get_edu method which returns Student education as reference type
   e. Create multiple instances of the struct.
   f. Call all the get methods of all the instances one by one and print them on your console.

ANSWER:

```
#[derive(Debug)]
struct Students<'a> {
    name: &'a str,
    age: &'a str,
    education: &'a str,
    timing: &'a str
}

impl <'a> Students <'a> {
    fn get_name(&self) -> &'a str {
        self.name
    }
    fn get_timing(&self) -> &'a str {
```

```
        self.timing
    }
    fn get_edu(&self) -> &'a str {
        self.education
    }
}

fn main() {
    let naeem = Students {
        name: "Naeem",
        age: "28",
        education: "Master",
        timing: "09-13"
    };
    let salman = Students {
        name: "Salman",
        age: "20",
        education: "Bachelor",
        timing: "13-18"
    };
    let haidar = Students {
        name: "Haidar",
        age: "24",
        education: "Bachelor",
        timing: "13-18"
    };
    println!("{:#?}", naeem.get_name());
    println!("{:#?}", naeem.get_timing());
    println!("{:#?}", naeem.get_edu());
    println!("{:#?}", salman.get_name());
    println!("{:#?}", salman.get_timing());
    println!("{:#?}", salman.get_edu());
    println!("{:#?}", haidar.get_name());
    println!("{:#?}", haidar.get_timing());
    println!("{:#?}", haidar.get_edu());
}
```

```rust
//a. Define a struct IOT_student with attributes (name, age, education).
#[derive(Debug)]
struct IOT_student {
    name: String,
    age: i32,
    education: String
}

//b. Define another struct IOT_instructor (name, age).
#[derive(Debug)]
struct IOT_instructor {
    name: String,
    age: i32
}

//c. Define a trait Questions with method ask_Questions with a default
//implementation which prints (“Zoom session will be LIVE, Zoom recording will
//    not be available. Quarter 2 studio recorded videos are available on Portal.”).
trait Questions {
    fn ask_questions(&self, name: String) {
        println!("Zoom session will be LIVE, Zoom recording will not be available.
        Quarter 2 studio recorded videos are available on Portal.")
    }
}
//d. Impl trait Questions for IOT_instructor which overrides the default implementation
//of method ask_question, takes student name as a parameter and prints on
//screen (“{} In case of any issue email to education@piaic.org”).
impl Questions for IOT_instructor {
    fn ask_questions(&self, name: String) {
        println!("{:?} In case of any issue email to education@piaic.org", name);
    }
}

//e. Create instances of both the structs and call Method ask_question.

fn main() {
    let naeem = IOT_student {
        name: String::from("Naeem"),
        age: 28,
        education: String::from("Master")
    };
    let sir_imran = IOT_instructor {
        name: String::from("Sir Imran"),
        age: 35
    };
    sir_imran.ask_questions(naeem.name);
}
```

2. Go through the solution of the largest function given at the end of 10.2 in the book and rewrite the solution but this time returning the smallest item instead largest.

```rust
//4. Go through the solution of the largest function given at the end of 10.2 in the book and
//rewrite the solution but this time returning the smallest item instead largest.
fn smallest<T: PartialOrd + Copy>(list: &[T]) -> T {
    let mut smallest = list[0];

    for &item in list {
        if item < smallest {
            smallest = item;
        }
    }
    smallest
}

fn main() {
    let number_list = vec![34, 50, 25, 100, 65];

    let result = smallest(&number_list);
    println!("The smallest number is {}", result);

    let char_list = vec!['y', 'm', 'a', 'q'];

    let result = smallest(&char_list);
    println!("The smallest char is {}", result);
}
```