1. Explain the effect of the panic! macro

Panic! ends program execution and unwinds and cleans up the stack. This includes releasing resources held by all frames in the stack in an orderly fashion, and therefore all resources held by the program.

2. Explain when panic! or Result/Option should be used and why, use examples

Panic!() is used when an unrecoverable error is encountered. An error is unrecoverable if attempting to continue execution will lead the program into an invalid state. In this case, continuing program execution will not only produce incorrect results but potentially create security vulnerabilities. The primary use case for manual panic! calls are function contract violations (user processes providing invalid input values).

Result and Option should be used when errors can be expected to occur occasionally, and those errors do not cause invalid state. In such cases, it is expected that the program recover gracefully, propagate the error for handling, and continue execution. Examples include: file operation errors, connectivity errors, and user input errors.

The following function prototypes will be used for the remaining questions:

fn open(filename: &str) -> Result<File, io::Error>

fn close(file: File) -> Result<(), io::Error>

fn hash(file: &File) -> Result<i32, math::Error>

3. Complete the function below to panic if the file doesn’t exist.

fn open\_or\_panic(filename: &str) -> File {

open(filename).unwrap()

}

4. Complete the function below to open the given file, calculate its hash, close the file, and return the result of the hash. If an error is encountered at any point, return None.

fn calculate\_hash(filename: &str) -> Option<i32> {

let file: File = open(filename)?;

let hash\_calc = hash(&file)?;

close(file)?;

hash\_calc

}

5. Write a function which takes a filename str reference as an argument, prints the files hash if it exists, and otherwise prints a relevant error message.

fn display\_hash(filename: &str) {

match open(filename) {

Ok(file) => match hash(file) {

Ok(hash) => println!(“Your hash is {}!”, hash),

Err(e) => println!(“An error occurred while calculating the hash!”),

},

Err(e) => println!(“An error occurred while opening the file!”),

};

}