Standard Library functions:

let exit(code: int) -> never;

/*The exit function that is part of the system will allow the program to prematurely top functioning within the vm whenever the function is called. The only parameter the exit function will take in will be an int. The code of this exit function will signify whether the program terminated gracefully or if some error occurred. Non-zero codes indicate errors. The return type of this function is never, which is a function that never resolves to return any value at all, similar to how rust's exit function behaves.*/

let prints(message: string) -> unit;

/*The prints() function will take in a string as its only parameter and output the string onto the console, without including a "\n" escape character at the end of the string to be printed. This function's return type is unit. If this function wanted to work with integers as well, then the integer value has to be converted into a string instead.*/

let printsln(message: string) -> unit;

/*The io function of printsln() will take in a string as its parameters and outputs the string to the console window. Similar to the prins() function, except at the end of the outputted string, this function adds a "\n" at the end of the inputted argument. This function's return type is unit. If this function wanted to work with integers as well, then the integer value has to be converted into a string instead. Its return type is unit.*/

let printi(value: int) -> unit;

/* The io function of printi() will take in a value of type int as its parameters and output the integer value to the console window. Unlike the prints() function, this function will only work with integer values instead. Parsing an integer value to a string will not work with this function. Its return type is unit.*/

let printiln(value: int) -> unit;

/*The io function of printiln() will take in a value of type int as its parameters and output the integer value to the console window. Unlike the printsln() function, this function will only work with integer values instead. Parsing an integer value to a string will not work with this function. This function will add a "\n" escape character to the end of the outputted integer value. Its return type is unit.*/

let asks(prompt: string) -> string;

/* The io function of akss() will print out a string prompt to the console window that will inform the user of a prompt, and the user would have to type into the console window a string value that needs to be entered. The return type of this function is a string, which needs to be stored in a corresponding variable or be used elsewhere. This is one of the console input methods that can be utilized, similar to how user input prompts in python work. */

let aski(prompt: string) -> int

/*The io funcion of aski() will print out a string prompt to the console window that will inform the user of a prompt, and the user would have to type into the console window a value that can be parsed into an int since all inputs and outputs from the console are treated as strings. Behind the scenes of this function, it converts the value entered into an integer. The return type of this function is an int, which would need to be stored into a corresponding variable to that type. This is another console input methods that can be used, similar to how input prompts in python work. If the user inputted value cannot be parsed to an integer, the function will call upon the exit function, passing in a non-zero code signaling the program to terminate gracefully. Additionally, inputs that are empty strings cannot be parsed and therefore the function will call upon the exit function, tell what the error is, and exit gracefully.*/