Question:-

ATM Simulator: Design a program that simulates an ATM machine, where users can withdraw and deposit money, view their balance, and change their PIN.

Answer:-#include <stdio.h>

```
// Function to withdraw money from account
Void withdraw(float *balance) {
   Float amount;
   Printf(“Enter the amount to withdraw: “);
   Scanf(“%f”, &amount);
   If (amount > *balance) {
      Printf(“Insufficient balance\n”);
   } else {
      *balance -= amount;
      Printf(“Transaction successful\n”);
   }
}

// Function to deposit money into account
Void deposit(float *balance) {
   Float amount;
   Printf(“Enter the amount to deposit: “);
   Scanf(“%f”, &amount);
   *balance += amount;
   Printf(“Transaction successful\n”);
}

// Function to view account balance
Void view_balance(float balance) {
```

```
    Printf(“Your balance is: %.2f\n”, balance);
}

// Function to change PIN
Void change_pin(int *pin) {
    Int new_pin;
    Printf(“Enter your new PIN: “);
    Scanf(“%d”, &new_pin);
    *pin = new_pin;
    Printf(“PIN changed successfully\n”);
}

Int main() {
    Float balance = 1000.0;
    Int pin = 1234;
    Int choice, entered_pin;

    Printf(“Welcome to the ATM\n”);
    Printf(“Please enter your PIN: “);
    Scanf(“%d”, &entered_pin);

    // Check if entered PIN is correct
    If (entered_pin != pin) {
        Printf(“Invalid PIN\n”);
        Return 0;
    }

    // Display menu
    While (1) {
```

```
Printf(“1. Withdraw\n”);
Printf(“2. Deposit\n”);
Printf(“3. View balance\n”);
Printf(“4. Change PIN\n”);
Printf(“5. Exit\n”);

Printf(“Enter your choice: “);
Scanf(“%d”, &choice);

Switch (choice) {
   Case 1:
      Withdraw(&balance);
      Break;
   Case 2:
      Deposit(&balance);
      Break;
   Case 3:
      View_balance(balance);
      Break;
   Case 4:
      Change_pin(&pin);
      Break;
   Case 5:
      Printf(“Thank you for using the ATM\n”);
      Return 0;
   Default:
      Printf(“Invalid choice\n”);
      Break;
}
```

```
    }


    Return 0;
}
```

Output :-

Welcome to the ATM

Please enter your PIN:

1.VARIABLES:-

In the C programming language, a variable is a named memory location that is used to store data of a specific type. Variables in C must be declared before they can be used, which means that the programmer must specify the variable's data type and name.

2.FUNCTION:-

In C, a function is a block of code that performs a specific task. It is used to break down a program into smaller, more manageable parts, making it easier to read, write, and maintain.

Functions in C have a return type, a name, and a set of parameters. The return type specifies the data type of the value that the function returns. The name is used to identify the function, and the parameters are used to pass data into the function.

3.FILES:-

In C programming, a file is a collection of data that is stored on a storage device like a hard disk or a flash drive. C provides several functions to work with files, including functions for creating, opening, reading, writing, and closing files.

The two main types of files In C are text files and binary files. Text files contain printable characters, such as letters, digits, and punctuation marks, while binary files contain non-printable characters, such as binary data and program code.

4.DATASETS:-

In C programming, a data set refers to a collection of related data values that are stored in memory. Data sets can be of different types, such as arrays, structures, or linked lists, and can contain data values of different data types, such as integers, characters, or floating-point numbers.

A data set can be used to represent a wide range of information, such as a list of names, a set of numerical values, or a collection of records that represent information about people, products, or transactions.

5.FEATURES:-

Procedural programming: C is a procedural language, which means that it allows developers to write programs as a sequence of instructions that execute one after another. This makes it easy to organize code and modularize functionality.

Low-level memory manipulation: C provides direct access to memory and allows developers to manipulate memory at a low level. This makes it possible to write highly optimized code that can be tailored to specific hardware architectures.

Structured programming: C supports structured programming constructs such as loops, conditionals, and functions, which make it easier to write code that is easy to read and maintain

6.SCREENSHOTS:-

C programming does not provide built-in functionality for capturing screenshots of a computer screen. However, there are external libraries and APIs that can be used to achieve this.

One popular library for taking screenshots in C programming is the Xlib library, which is a low-level library for interacting with the X Window System used on Unix-like operating systems. Here is an example of how to take a screenshot of the entire screen