Create a finite state machine (FSM) in C for a simple vending machine. The machine should have states like "Idle," "Selecting Product," "Processing Payment," and "Dispensing Product." Provide the state transition logic based on user inputs

**Answer:**

#include <stdio.h> typedef enum {

IDLE, SELECTING\_PRODUCT, PROCESSING\_PAYMENT, DISPENSING\_PRODUCT

} VendingState; // we can use this vending state like data types (int ,char)

//we defining product price here #define PRODUCT\_PRICE 50

void vending\_machine() {

VendingState currentState = IDLE; //dtype var =val int userInput, payment = 0;

//swtch starts here

while (1) {

switch (currentState) { case IDLE:

printf("\nVending Machine is IDLE. Please select a product (1 to continue, 0 to exit): ");

//if user inpt is 1 it executes otherwise it exits scanf("%d", &userInput);

if (userInput == 1) {

currentState = SELECTING\_PRODUCT;//assigning to selpro

} else {

printf("Exiting...\n");

return;

}

break;

case SELECTING\_PRODUCT:

//after user input is 1 the program will execute from here

printf("Product selected. Price is %d units. Proceed to payment? (1 for Yes, 0 for No): ",

PRODUCT\_PRICE);//pro price is macro defined scanf("%d", &userInput);

if (userInput == 1) {

currentState = PROCESSING\_PAYMENT;

} else {

currentState = IDLE;

}

break;

case PROCESSING\_PAYMENT://once user payed 50 only it goes to another step till that it waits for the paypment

printf("Insert payment (current total: %d): ", payment); scanf("%d", &userInput);

payment += userInput;

if (payment >= PRODUCT\_PRICE) { printf("Payment received. Dispensing product...\n"); currentState = DISPENSING\_PRODUCT;

} else {

printf("Insufficient payment to the product that you are going to purchase. Please add

more.\n");

}

break;

case DISPENSING\_PRODUCT:

printf("Product dispensed. Thank you!\n"); payment = 0;

currentState = IDLE; break;

default:

printf("Unknown state! brother\n"); return;

}

}

}

//-------------main function here-------- int main() {

vending\_machine(); return 0;

}

Explain the concept of a super loop in C programming. Write a simple C program that continuously monitors three different flags in a super loop and performs an action when a specific flag is set

**Answer**

#include <stdio.h> #include <stdbool.h> #include <unistd.h>

bool flag1 = false; bool flag2 = false; bool flag3 = false;

void setFlags() {

static int count = 0; count++;

if (count % 5 == 0) flag1 = true; if (count % 10 == 0) flag2 = true; if (count % 15 == 0) flag3 = true;

}

// Main function int main() {

while (true) { // Super loop setFlags();

if (flag1) {

printf("Action for Flag 1 triggered!\n"); flag1 = false; // Reset flag after action

}

if (flag2) {

printf("Action for Flag 2 triggered!\n"); flag2 = false; // Reset flag after action

}

if (flag3) {

printf("Action for Flag 3 triggered!\n"); flag3 = false; // Reset flag after action

}

sleep(1); // Sleep for 1 second

}

return 0; //no end for this program it will reach this line

}