```
//=============================================================================================================//
//                                                                                                             //
//                                          Atributos de classe                                                //
//                                                                                                             //
//=============================================================================================================//

//Entradas digitais
int sensorPortaEsquerdaAberta = 22;
int sensorPortaDireitaAberta = 23;
int sensorPortaEsquerdaFechada = 24;
int sensorPortaDireitaFechada = 25;
int sensorAndar1 = 26;
int sensorAndar2 = 27;
int botaoChamadaCima = 28;
int botaoChamadaBaixo = 29;
int botaoCabineEmergencia = 30;
int botaoCabineBaixo = 31;
int botaoCabineCima = 32;
int botaoAbrePorta = 33;
int botaoFechaPorta = 34;

//Saídas digitais

int habilitaMotorPortaEsquerda = 45;
int habilitaMotorPortaDireita = 46;
int abrePortaEsquerdaA = 47; //Para abrir setar em 1, para fechar setar em 0
int abrePortaEsquerdaB = 48; //Para abrir setar em 0, para fechar setar em 1
int abrePortaDireitaA = 49;
int abrePortaDireitaB = 50;
int motorSobe = 51;
int motorDesce = 52;
int motorPara = 53;

//Entradas polarizadas (lógica no método setup)z
boolean sensorPEA_EP;
boolean sensorPEF_EP;
boolean sensorPDA_EP;
boolean sensorPDF_EP;



//=============================================================================================================//
//                                                                                                             //
//                                              Método setup                                                   //
//                                                                                                             //
//=============================================================================================================//

void setup() {
  //Variáveis de classe

  //Seta os pinos como entradas digitais e seta-os como falso.
  for (int i = 22; i <= 34; i++) {
    pinMode(i, INPUT);
    digitalWrite(i, LOW);
  }

  //Seta os pinos como saídas digitais e seta-os como falso.
  for (int i = 45; i <= 53; i++) {
    pinMode(i, OUTPUT);
    digitalWrite(i, LOW);
  }
  Serial.begin(9600);
  int matrizSensores[] = {22, 23, 24, 25, 26, 27};
  int matrizBotoes[] = {28, 29, 30, 31, 32, 33, 34};

}
//=============================================================================================================//
//                                                                                                             //
//                                              Método polariza                                                //
//                                                                                                             //
//=============================================================================================================//

void polariza() {

  delay(1000);

  //Exibe o status de cada sensor - físico
  Serial.print("Esquerda aberta= " + String(digitalRead(sensorPortaEsquerdaAberta)) + " ");
  Serial.print("Esquerda fechada= " + String(digitalRead(sensorPortaEsquerdaFechada)) + " ");
  Serial.print("Direita aberta= " + String(digitalRead(sensorPortaDireitaAberta)) + " ");
  Serial.println("Direita fechada= " + String(digitalRead(sensorPortaDireitaFechada)) + " ");

  //Exibe o status de cada sensor - polarizado
  Serial.print("Esquerda aberta= " + String(digitalRead(sensorPEA_EP)) + " ");
  Serial.print("Esquerda fechada= " + String(digitalRead(sensorPEF_EP)) + " ");
  Serial.print("Direita aberta= " + String(digitalRead(sensorPDA_EP)) + " ");
  Serial.println("Direita fechada= " + String(digitalRead(sensorPDF_EP)) + " ");


  //Lógica de polarização de entradas
  //Sensores das portas

  //Sensor de porta totalmente aberta. Este sensor é NF. O ponto esperado é 1. Quando comuta para 0 ele foi pressionado.
  sensorPEA_EP = digitalRead(sensorPortaEsquerdaAberta) ^ 1;
  //Sensor de porta totalmente aberta. Este sensor é NF. O ponto esperado é 1. Quando comuta para 0 ele foi pressionado.
  sensorPDA_EP = digitalRead(sensorPortaDireitaAberta) ^ 1;
  //Sensor de porta totalmente fechada. Este sensor é NA. O ponto esperado é 0. Quando comuta para 1 ele foi pressionado.
  sensorPEF_EP = digitalRead(sensorPortaEsquerdaFechada) ^ 0;
  //Sensor de porta totalmente fechada. Este sensor é NA. O ponto esperado é 0. Quando comuta para 1 ele foi pressionado.
```

```
    sensorPDF_EP = digitalRead(sensorPortaDireitaFechada) ^ 0;
  }

  //=============================================================================================================//
  //                                                                                                             //
  //                                          Método abrePortas                                                  //
  //                                                                                                             //
  //=============================================================================================================//
  void abrePortas(String lado) {

    //Escreve no monitor qual porta foi pedida para ser aberta
    String str = "Pedido de abertura da porta " + lado;
    Serial.println(str);

    //Lê os sensores fim-de-curso
    //Primeiro testa se a porta solicitada foi a da esquerda
    if (lado.equals("Esquerda")) {
      //Agora verifica se a porta já se encontra aberta
      if (sensorPEA_EP) {
        Serial.println("Porta esquerda totalmente aberta");
        //Certifica-se de parar o motor de abertura da porta esquerda
        digitalWrite(abrePortaEsquerdaA, LOW); //Retira pulso para abertura
        digitalWrite(abrePortaEsquerdaB, LOW); //Retira pulso para abertura
        digitalWrite(habilitaMotorPortaEsquerda, LOW); //Inibe ponte-h da porta da esquerda
      }
      else {
        //Se a porta da esquerda não se encontrar totalmente aberta o controlador solicita sua abertura
        Serial.println("Abrindo porta esquerda");
        //Certifica-se de mandar abrir a porta esquerda
        digitalWrite(abrePortaEsquerdaA, HIGH); //Ativa pulso para abertura
        digitalWrite(abrePortaEsquerdaB, LOW); //Retira pulso para abertura
        digitalWrite(habilitaMotorPortaEsquerda, HIGH); //Habilita ponte-h da porta da esquerda
      }

      //Depois testa se a porta solicitada foi a da direita
    } else   if (lado.equals("Direita")) {
      //Agora verifica se a porta já se encontra aberta
      if (sensorPDA_EP) {
        Serial.println("Porta direita totalmente aberta");
        //Certifica-se de parar o motor de abertura da porta esquerda
        digitalWrite(abrePortaDireitaA, LOW); //Retira pulso para abertura
        digitalWrite(abrePortaDireitaB, LOW); //Retira pulso para abertura
        digitalWrite(habilitaMotorPortaDireita, LOW); //Inibe ponte-h da porta da direita
      }
      else {
        //Se a porta da esquerda não se encontrar totalmente aberta o controlador solicita sua abertura
        Serial.println("Abrindo porta direita");
        //Certifica-se de mandar abrir a porta esquerda
        digitalWrite(abrePortaDireitaA, HIGH); //Ativa pulso para abertura
        digitalWrite(abrePortaDireitaB, LOW); //Retira pulso para abertura
        digitalWrite(habilitaMotorPortaDireita, HIGH); //Habilita ponte-h da porta da direita
      }
    }

  }

  //=============================================================================================================//
  //                                                                                                             //
  //                                          Método fechaPortas                                                 //
  //                                                                                                             //
  //=============================================================================================================//
  void fechaPortas(String lado) {

    //Escreve no monitor qual porta foi pedida para ser aberta
    String str = "Pedido de fechamento da porta " + lado;
    Serial.println(str);

    //Lê os sensores fim-de-curso
    //Primeiro testa se a porta solicitada foi a da esquerda
    if (lado.equals("Esquerda")) {
      //Agora verifica se a porta já se encontra fechada
      if (sensorPEF_EP) {
        Serial.println("Porta esquerda totalmente fechada");
        //Certifica-se de parar o motor de fechamento da porta esquerda
        digitalWrite(abrePortaEsquerdaA, LOW); //Retira pulso para fechamento
        digitalWrite(abrePortaEsquerdaA, LOW); //Retira pulso para fechamento
        digitalWrite(habilitaMotorPortaEsquerda, LOW); //Inibe ponte-h da porta da esquerda
      }
      else {
        //Se a porta da esquerda não se encontrar totalmente fechada o controlador solicita seu fechamento
        Serial.println("Fechando porta esquerda");
        //Certifica-se de mandar fechar a porta esquerda
        digitalWrite(abrePortaEsquerdaA, LOW); //Ativa pulso para fechamento
        digitalWrite(abrePortaEsquerdaB, HIGH); //Retira pulso para fechamento
        digitalWrite(habilitaMotorPortaEsquerda, HIGH); //Habilita ponte-h da porta da esquerda
      }

      //Depois testa se a porta solicitada foi a da direita
    } else   if (lado.equals("Direita")) {
      //Agora verifica se a porta já se encontra fechada
      if (sensorPDA_EP) {
        Serial.println("Porta direita totalmente fechada");
        //Certifica-se de parar o motor de abertura da porta esquerda
        digitalWrite(abrePortaDireitaA, LOW); //Retira pulso para abertura
        digitalWrite(abrePortaDireitaB, LOW); //Retira pulso para abertura
        digitalWrite(habilitaMotorPortaDireita, LOW); //Inibe ponte-h da porta da direita
      }
      else {
        //Se a porta da direita não se encontrar totalmente aberta o controlador solicita sua abertura
```

```
        Serial.println("Fechando porta direita");
        //Certifica-se de mandar abrir a porta esquerda
        digitalWrite(abrePortaDireitaA, LOW); //Ativa pulso para abertura
        digitalWrite(abrePortaDireitaB, HIGH); //Retira pulso para abertura
        digitalWrite(habilitaMotorPortaDireita, HIGH); //Habilita ponte-h da porta da direita
      }
    }

  }

  //============================================================================================//
  //                                                                                            //
  //                                        Método botoes                                       //
  //                                                                                            //
  //============================================================================================//

  void botoes() {


    //Botoes internos da cabine
    /*Abertura e fechamento das portas
      Se o botão de abertura da cabine for acionado ele solicita a abertura ou fechamento
      de ambas as portas até que os sensores sejam todos acionados.*/
    String lado;
    if (digitalRead(botaoAbrePorta)) {
      Serial.println("Botao de abertura aciondao - cabine");
      do {
        lado = "Esquerda";
        abrePortas(lado);
        lado = "Direita";
        abrePortas(lado);
        polariza();
      }
      while (!sensorPEA_EP && !sensorPDA_EP);
    }

    else if (digitalRead(botaoFechaPorta)) {
      Serial.println("Botao de fechamento aciondao - cabine");
      do {
        lado = "Esquerda";
        fechaPortas(lado);
        lado = "Direita";
        fechaPortas(lado);
        polariza();
      }
      while (!sensorPEA_EP && !sensorPDA_EP);
    }
    else {
      //Nenhum botão acionado. Os motores devem parar e deixar o sistema automático os posicionar
      Serial.println("Nao ha pedido de abertura ou fechamento");
      digitalWrite(abrePortaEsquerdaA, LOW);
      digitalWrite(abrePortaEsquerdaB, LOW);
      digitalWrite(abrePortaDireitaA, LOW);
      digitalWrite(abrePortaDireitaB, LOW);
      digitalWrite(habilitaMotorPortaEsquerda, LOW); //Inibe ponte-h da porta da esquerda
      digitalWrite(habilitaMotorPortaDireita, LOW); //Inibe ponte-h da porta da direita
    }
  }

  //============================================================================================//
  //                                                                                            //
  //                                      Método cronometro                                     //
  //                                                                                            //
  //============================================================================================//

  /* Este método retorna o tempo que um evento demorou, recebendo um valor de tempo como argumento e a
    condição do evento como boleano*/

  long cronometro(long tempo, boolean evento) {

    long tempoPassado;

    do {
      tempoPassado = millis() - tempo;
    }
    while (evento);

    return tempoPassado;

  }

  //============================================================================================//
  //                                                                                            //
  //                                         Método loop                                        //
  //                                                                                            //
  //============================================================================================//

  void loop() {

    do {

      //Exibe no monitor o tempo em que o programa está rodando
      Serial.println("-----------------------------------------------------------------");
      Serial.println(String((millis() / 1000)) + " segundos ativo" );
      int tempo = 0;

      polariza();
      botoes();
    }
```

```
298     while (Serial.available() > 0);
299 
300   }
301 
302 
303 
```