RELATÓRIO DO TRABALHO PRÁTICO INDIVIDUAL

Miguel Curto Castela
uc2022212972

14 DE NOVEMBRO DE 2022

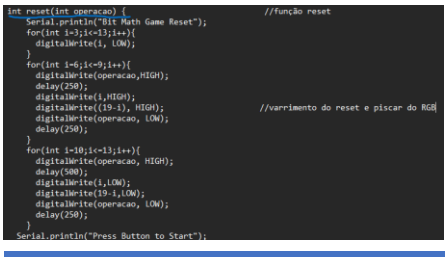
TECNOLOGIA DA INFORMÁTICA

O Código:

O código para este jogo centrou-se à volta de funções modulares, pois deste modo é possível relocalizar porções úteis do código, sem necessidade de haver repetição do mesmo. Usei diversas funções neste código; estas facilitam também a organização e o bom funcionamento deste. Passo a explicar as funções usadas no trabalho elaborado:

Função reset:

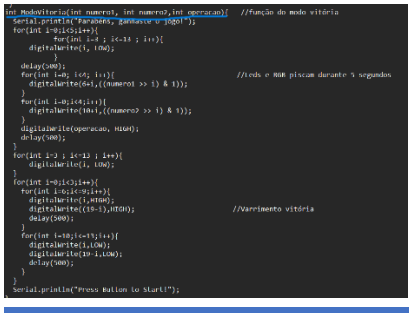
Nesta função são usados ciclos 'for' e delays de 250ms para fazer o varrimento desejado dos leds sempre que há um reset, assim como o piscar do led RGB enquanto isto decorre. É usado o Serial.println para imprimir a mensagem "Bit Math Game Reset" sempre que queremos dar reset no jogo.



Esta função é ativada quando o botão é pressionado durante mais do que um segundo:

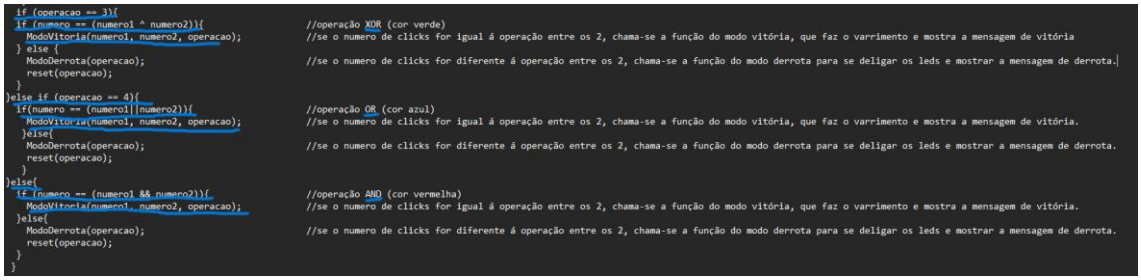
Função ModoVitória:

Esta função usa ciclos 'for' e 'delays' de 500ms para fazer o varrimento do modo vitória, são também usados ciclos 'for' para que os LEDs e o RGB pisquem durante 5 segundos, antes do varrimento, como desejado:



Esta função é ativada quando, para qualquer uma das funções logicas (XOR, OR ou AND), a variável número (número de vezes que o botão é pressionado) é igual ao resultado da operação lógica (random), representada pela cor do RGB. Esta operação lógica é aplicada entre o numero1 e o numero2 (gerados de maneira aleatória, de 1 a 16, já que com **quatro bits**, é possível criar 16 valores diferentes).

O 'if', o 'else if ' e o 'else' são usados quando o resultado da operação escolhida é igual:



Função ModoDerrota:

Nesta função são usados ciclos “for” para desligar os leds e o RGB após uma derrota, seguido do piscar do RGB, como desejado:

```
int ModoDerrota(int variavel){                    //função do modo derrota
  Serial.println("perdeste o jogo!");
  for (int i=6;i<=13;i++) {                       //desligar leds após derrota
    digitalWrite(i,LOW);
  }
  for(int i = 1;i<=5;i++) {
    digitalWrite(variavel,LOW);                   //piscar do RGB
    delay(500);
    digitalWrite(variavel,HIGH);
    delay(500);
  }
  Serial.println("Press Button to Start!");
}
```

Esta função é ativada quando, para qualquer uma das funções logicas (XOR, OR ou AND), a variável número (número de vezes que o botão é pressionado) é diferente ao resultado da operação lógica (random), representada pela cor do RGB. Esta operação lógica é aplicada entre o numero1 e o numero2 (gerados de maneira aleatória, de 1 a 16, já que com **quatro bits**, é possível criar 16 valores diferentes).

O ‘else’ é usado quando o resultado da operação escolhida é diferente:

```
 if (operacao == 3){
  if (numero == (numero1 ^ numero2)){          //operação XOR (cor verde)
    ModoVitoria(numero1, numero2, operacao);   //se o numero de clicks for igual á operação entre os 2, chama-se a função do modo vitória, que faz o varrimento e mostra a mensagem de vitória
  } else {
    ModoDerrota(operacao);                     //se o numero de clicks for diferente á operação entre os 2, chama-se a função do modo derrota para se deligar os leds e mostrar a mensagem de derrota.
    reset(operacao);
  }
}else if (operacao == 4){
  if(numero == (numero1||numero2)){            //operação OR (cor azul)
    ModoVitoria(numero1, numero2, operacao);   //se o numero de clicks for igual á operação entre os 2, chama-se a função do modo vitória, que faz o varrimento e mostra a mensagem de vitória.
  }else{
    ModoDerrota(operacao);                     //se o numero de clicks for diferente á operação entre os 2, chama-se a função do modo derrota para se deligar os leds e mostrar a mensagem de derrota.
    reset(operacao);
  }
}else{
  if (numero == (numero1 && numero2)){         //operação AND (cor vermelha)
    ModoVitoria(numero1, numero2, operacao);   //se o numero de clicks for igual á operação entre os 2, chama-se a função do modo vitória, que faz o varrimento e mostra a mensagem de vitória.
  }else{
    ModoDerrota(operacao);                     //se o numero de clicks for diferente á operação entre os 2, chama-se a função do modo derrota para se deligar os leds e mostrar a mensagem de derrota.
    reset(operacao);
  }
}
```

Principais dificuldades:

Ao longo da resolução deste trabalho tive 2 dificuldades principais:

- Na implementação do while, usando o millis(), tanto na duração de 1 jogo (7000ms) ou nas mensagens que vão aparecendo em cada jogo (50% do tempo passado ou 75%), resolvi este problema usando uma variável (instant) para resolver os erros nas mensagens de tempo.

- No debounce do primeiro click, resolvi este problema ao implementar um segundo debounce no início do loop.

Superei estes desafios após a testagem continua do meu código e da resolução de erros ao longo deste.

O que aprendi:

Com este trabalho, aprendi algumas funcionalidades do arduino que não sabia implementar nesta escala até agora (como os ciclos for, e o while).

Aprendi também a construir um código organizado, com um número considerável de funções e variáveis, e a solucionar melhor os erros que me apareceram no código ao longo da sua implementação.

Dúvidas que surgiram ao longo do trabalho:

- Após a função reset ser chamada (a partir de um click com duração superior a 1 segundo), a função derrota também é ativada. Não tem implicações nenhumas no funcionamento normal do jogo, mas eu tentei solucionar o problema mudando a ordem das funções, mas sem resultados positivos.
- Numa versão mais incial do código, para meter os leds em input eu tinha usado uma função como a seguinte: for (led = 3; led < 14; led++) {pinMode(led, OUTPUT);}, e tinha definido apenas uma variável led, antes do setup. Isto criava erros, e, consequentemente, os leds não ligavam. Eu resolvi este problema, como é possível verificar no meu código, criando variáveis individuais para todos os leds e para todos os inputs. No entanto, não percebo porque é que o primeiro método não resultaria.
- Para o led RGB funcionar corretamente, tive de ativar a função reset depois da função ModoDerrota ser ativada. Isto não tem implicações nenhumas no funcionamento normal do jogo, mas não percebo o porquê de ocorrer. Se eu não ativasse o reset depois do ModoDerrota, o loop não funcionaria da mesma maneira.