IOT 프로그래밍

랜덤 숫자 암산 게임

6조-정윤정, 유정식, 정아영

목차

1. 게임 설명

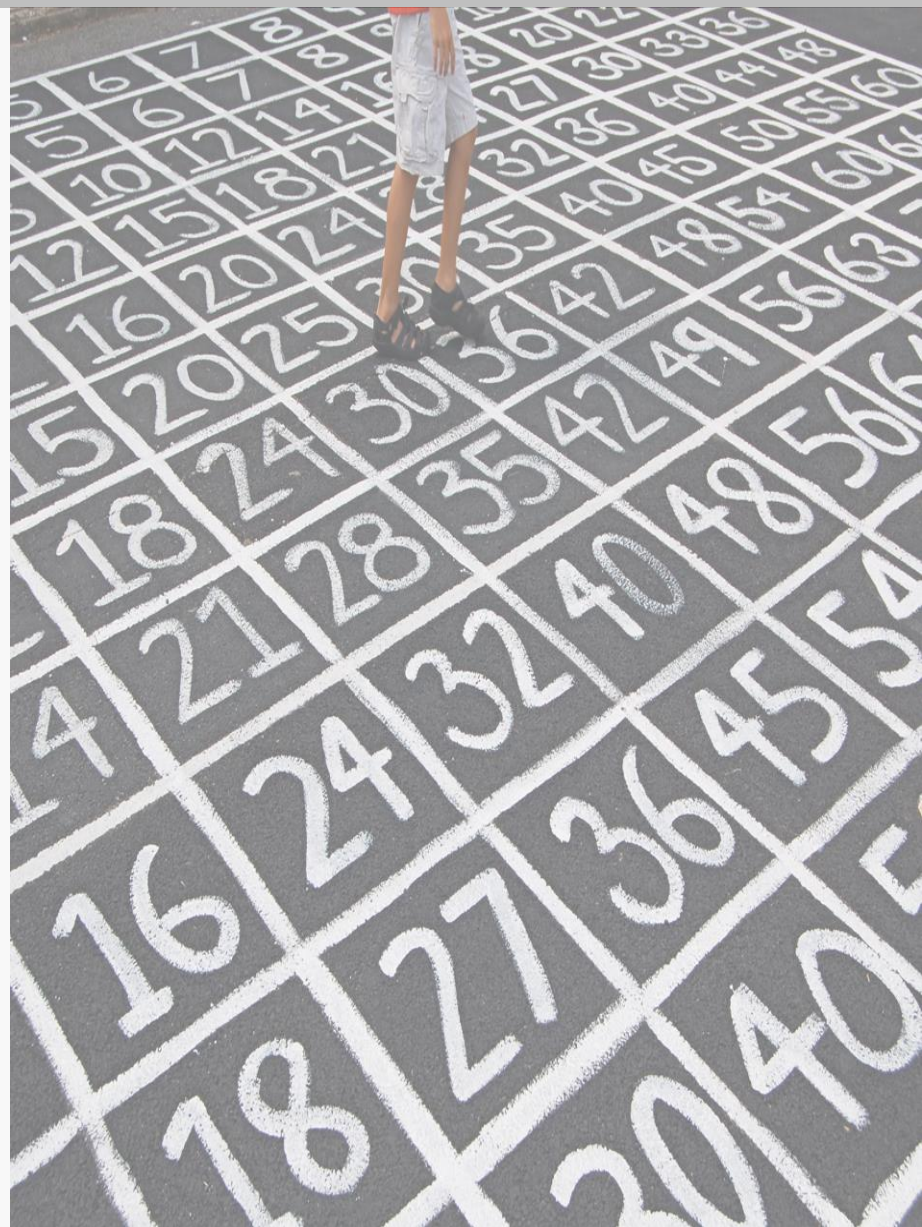
- **게임 내용**

  사용자가 선택한 범위의 숫자가 사용자가 선택한 개수만큼 CLCD에 출력되면 사용자는 숫자와 숫자 사이에 랜덤으로 출력되는 연산자에 따라 암산을 진행하고, 답을 입력한다.

  시스템은 사용자의 답을 입력받아 채점 후 결과를 알려주고, 한 세트 완료 후 점수를 출력한다.

- **게임 규칙**

  사용자가 입력가능한 숫자 범위는 1 ~ 99, 숫자 개수의 범위는 0 ~ 9 이며, 이 범위를 벗어나는 경우 게임은 시작되지 않는다.



2. 수정사항

- **입출력 장치**

  입력 – Tact Switch 사용
  출력 – Character LCD 사용

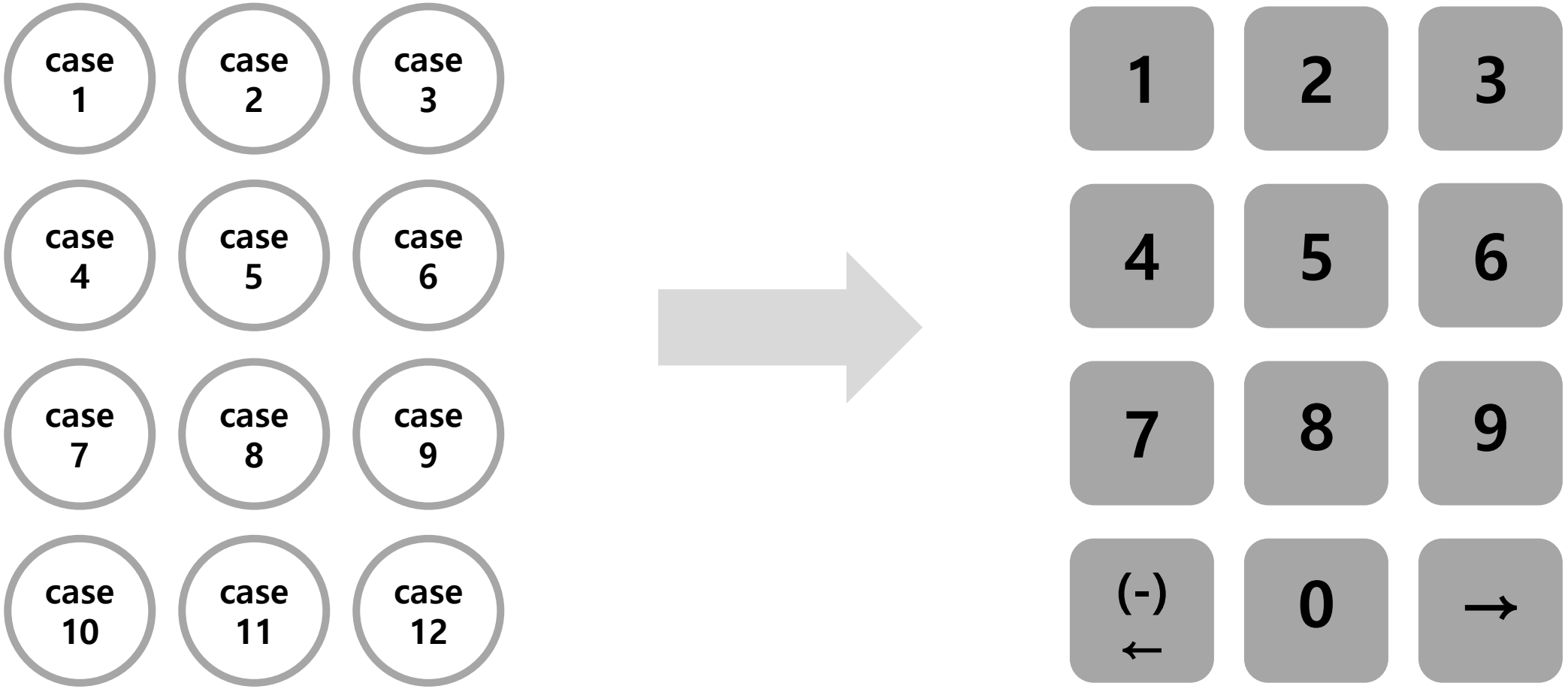
- **게임 규칙**

  사용자가 입력한 출력 숫자의 최댓값에 따라 숫자의 출력 속도 변화

  → 출력 속도 고정

3. 알고리즘 및 코드

입력 장치 – Tact Switch



3. 알고리즘 및 코드

1) 사용할 숫자의 최댓값 입력받기

case 1) 1~9 이외의 버튼을 클릭

```
if ((count[user_max_input] == 10 || count[user_max_input] == 11) && user_max_input == 0) {
          // 숫자입력없이 바로 확인버튼을 눌렀을때 오류처리
          clcd_input("input number!!!");
          user_max_input = user_max_input - 1;
          }
```

case 2) 입력받은 숫자의 최댓값이 한자리

```
else if (count[user_max_input] == 11 && user_max_input == 1)
          {// 확인 버튼 누르면 함수 탈출
          count[1] = count[0];
          // 입력받은 숫자를 count[1]에 저장해서 아래서 사용
          number = count[1];// 한자리의 숫자 출력
          countLCD1(number);
          usleep(2000 * 1000);
          break;
          }
```

case 3) 입력받은 숫자의 최댓값이 두자리

```
else if (count[user_max_input] == 11 && user_max_input == 2)
          {
          count[0] = count[0] * 10;// 두자리 숫자 출력
          count[1] = count[0] + count[1];
          number = count[1];
          usleep(2000 * 1000);
          countLCD1(number);
          break;
          }
```

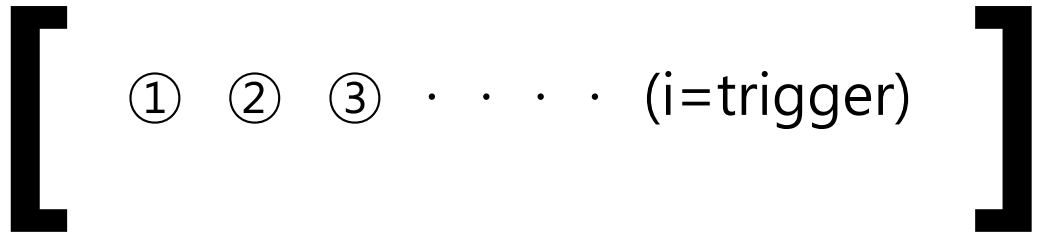
3. 알고리즘 및 코드

2) 사용할 랜덤 숫자와 연산자 저장

랜덤 숫자 저장

```
int data[trigger];

srand((int)time(NULL));// 랜덤숫자 배열로 저장
for (i = 0; i < trigger; i++) {
data[i] = rand() % count[1] + 1;//data[i]는 숫자 저장
}
```

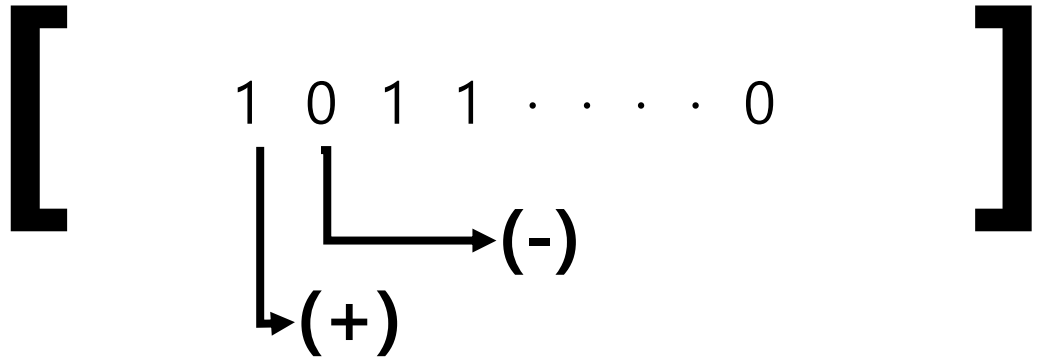
data



연산자 저장

```
int op2[trigger - 1];

for (i = 0; i < trigger - 1; i++) {
op2[i] = rand() % 2;// op2는 부호 저장
}
```

op2



3. 알고리즘 및 코드

3) 프로그램 내에서 정답 계산, 숫자와 연산자 출력

채점을 위한 정답 계산

```
while (1) {// 시스템상 정답 계산
          if (op2[i-1] == 1 && (i-1) >= 0) {// 1 일때가 +
          ans = ans + data[i];
          }
          else if (op2[i-1] == 0 && (i-1) >= 0) {// 0 일때가 -
          ans = ans - data[i];
          }

          if (i == (trigger - 1)) {
          break;
          }
          i += 1;
     }
```

숫자와 연산자 번갈아가며 출력

```
number = data[i];
countLCD3(number);// 임의의 숫자 출력
usleep(2000 * 1000);

if (op2[i] == 1) {// 임의의 부호 출력
          if (i == (trigger-1)) {
                    break;
          }
          clcd_input(" + ");
          usleep(2000 * 1000);
}
else {
          if (i == (trigger-1)) {
                    break;
          }
          clcd_input(" - ");
          usleep(2000 * 1000);
}
i += 1;
```

3. 알고리즘 및 코드

4) 사용자의 답 입력받기

- 답을 입력하지 않고 확인버튼을 누른 경우

```
if (counting[0] == 11) {// 숫자입력없이 바로 확인버튼을 눌렀을때 오류처리
          clcd_input("input number!!!");
          count_input_num = count_input_num - 1;
}
```

- 음수, 지우기 버튼을 10번 버튼에 동시 할당

```
else if (counting[0] == 10 && mark == 0) {// mark가 1일때 음수
mark = 1;
count_input_num = count_input_num - 1;
}
else if (counting[0] == 10 && mark == 1) {// mark가 0일때 양수
mark = 0;
count_input_num = count_input_num - 1;
}
else if (count_input_num >= 1 && counting[count_input_num] == 10) {
// 지우기 버튼
counting[count_input_num - 1] = 0;
count_input_num = count_input_num - 2;
}
```

- 답의 자릿수에 대한 처리

게임 시작 시 숫자를 입력받는 알고리즘과 동일하게 처리

- 입력받은 답이 음수인 경우

```
if (mark == 1) {
          // 기존에 입력받은 값을 음수로 변환
          usr_ans = -usr_ans;
}
```

3. 알고리즘 및 코드

5) 채점 및 점수 출력

채점 및 점수 출력, FINISH

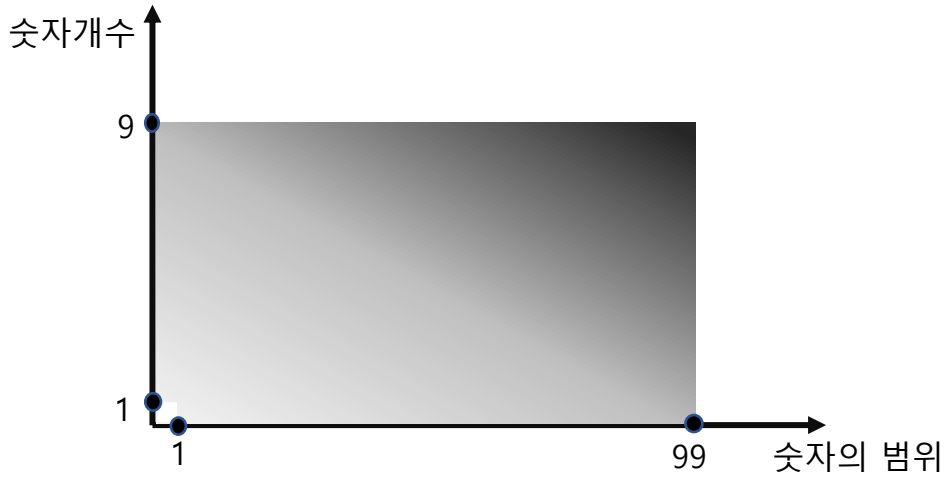
```
            if (ans == usr_ans) {
            lastscore = lastscore + 1;
            clcd_input("correct!");
            usleep(2000 * 1000);
            }
            else {
            clcd_input("wrong answer");
            usleep(2000 * 1000);
            }
            }
number = lastscore;
scoreLCD(number);
usleep(2000 * 1000);

clcd_input("FINISH");
return 0;
```

4. 차별점

- 게임 기능 측면

- 암산에 사용할 숫자의 범위와 개수를 사용자가 직접 설정 가능

→ 난이도 조절 가능



- 숫자만 랜덤으로 출력하는 것이 아니라 연산자도 랜덤으로 출력하여 좀 더 고차원적인 게임 진행 가능

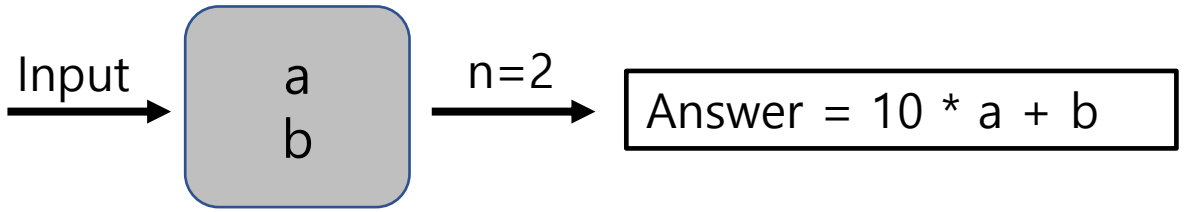
- 알고리즘 구현 측면

- 하나의 Tact Switch를 상황에 따라 두가지 기능으로 사용할 수 있도록 정보 할당

case10

(-)
←

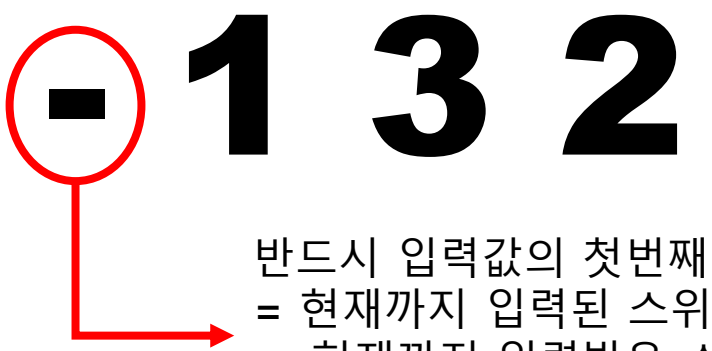
- 사용자가 입력하는 Tact Switch개수에 따라 자릿수에 알맞은 수를 계산하여 입력받기



4. 차별점

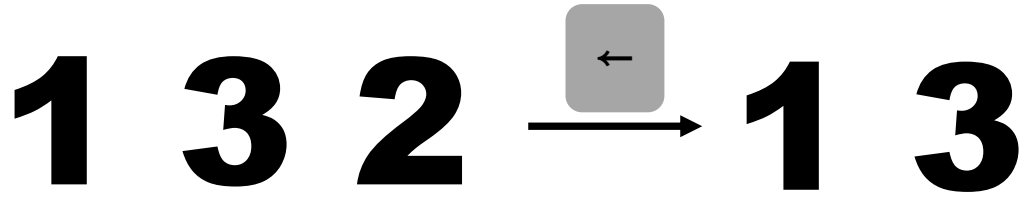
- 하나의 Tact Switch를 상황에 따라 두가지 기능으로 사용할 수 있도록 정보 할당

case 1) 음수 버튼 사용 (-)



반드시 입력값의 첫번째에서 사용
= 현재까지 입력된 스위치의 개수가 0일 때만 사용
→ 현재까지 입력받은 스위치의 개수가 0개일 때만 case10 버튼을 (-)부호로 할당

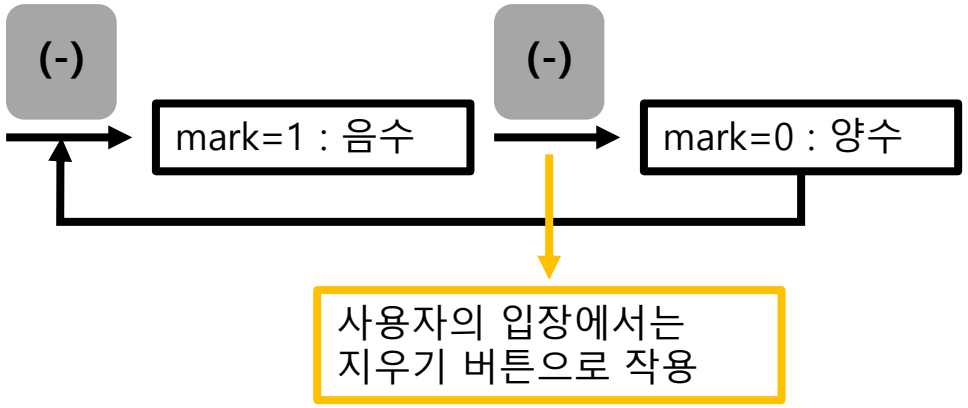
case 2) 지우기 버튼 사용

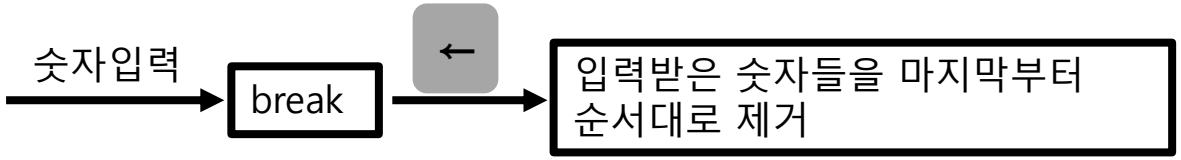


지우기 버튼은 입력된 스위치의 개수가 한 개 이상일 때만 사용
(현재까지 입력된 정보가 없으면 지울 정보가 없기때문)

4. 차별점

초기화
mark = 0 : 양수의 상태
count_input_num = 0 : 현재까지 입력된 숫자의 개수





숫자가 입력되면 프로그램은 mark값을 저장해 두었다가 사용자가 답안 입력 완료 버튼을 누르면 mark값에 따라 입력받은 답안을 양수 또는 음수로 출력

```
else if (counting[0] == 10 && mark == 0) {// mark가 1일때 음수
mark = 1;
count_input_num = count_input_num - 1;
}
else if (counting[0] == 10 && mark == 1) {// mark가 0일때 양수
mark = 0;
count_input_num = count_input_num - 1;
}
else if (count_input_num >= 1 && counting[count_input_num] == 10) {
// 지우기 버튼
counting[count_input_num - 1] = 0;
count_input_num = count_input_num - 2;
}
else if (counting[count_input_num] == 11 && count_input_num == 1) {
// 확인 버튼 누르면 함수 탈출
counting[1] = counting[0];
// 입력받은 숫자를 counting[1]에 저장해서 아래서 사용
number = counting[1];// 한자리 숫자 출력
usr_ans = number;
if (mark == 1) {
// 확인 버튼을 입력받고 mark가 1일때 입력받은 값을 음수로 전환
number = -number;
}
usransLCD(number);
usleep(2000 * 1000);
break;
```

5. 구현영상

https://www.youtube.com/watch?v=CR5eMvyP-oY

참고자료

https://syki66.github.io/blog/2020/06/15/H-smart4412TKU.html

https://hongci.tistory.com/92?category=219350

https://github.com/jinwoo1225/SnakeGameWithSmart4412/blob/main/README.md


THANK YOU