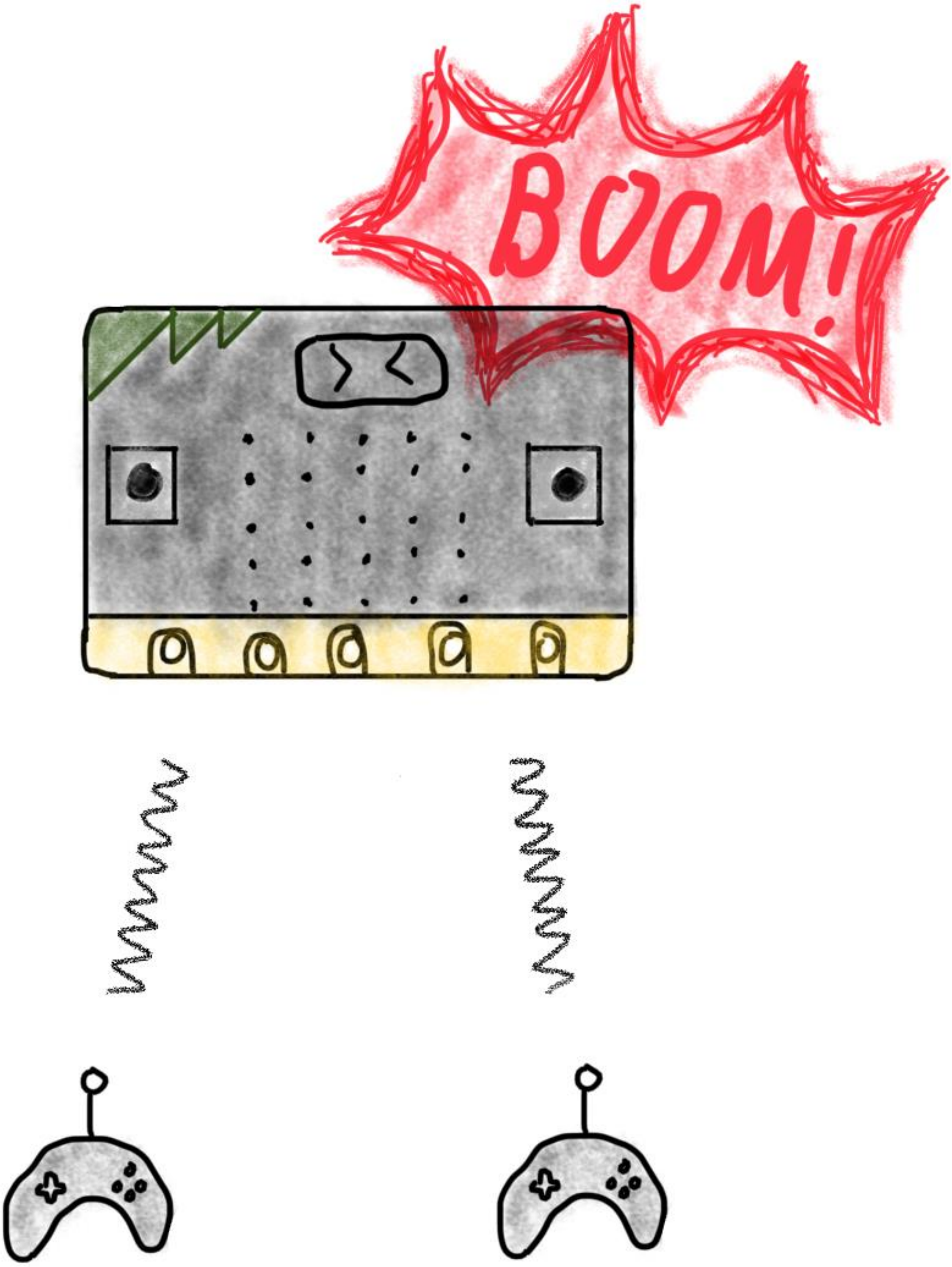
Stewards MKMCF Ma Ko Pan Memorial College

S.2 ICT Micro:bit Project

Bomberman Game



Content

Self-introduction

Hi! I am Paco from class 2D, I have made a microbit program for my ICT project “Bomberman Game”. Now I’m going to introduce the program.

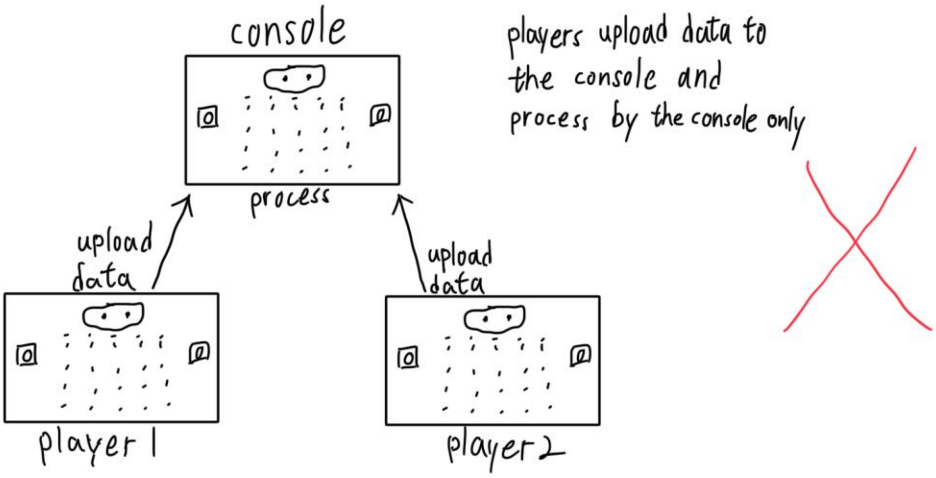
Game introduction

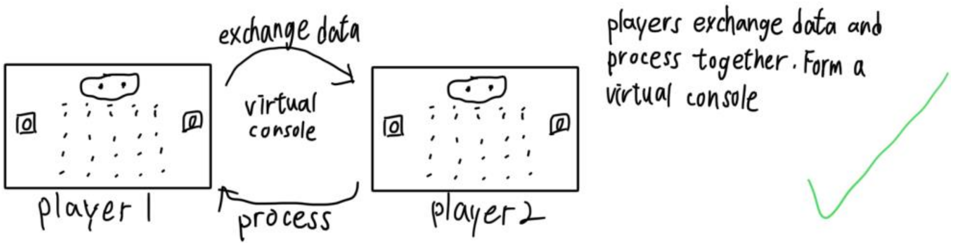
By the way before that, you should know what is bomberman game. That’s a little game we can play with our friends. First, there will be a bomb with a random number(10-25). Then a player will be random to play first. The players can enter number 1 to 3. Then is the next player’s turn. When the sum of the number that’s the players entered is higher or equal to the bomb, the game will end. The player that’s entered the last number will be the loser. It is an interesting game! Now let’s have a look of the program!

Program introduction

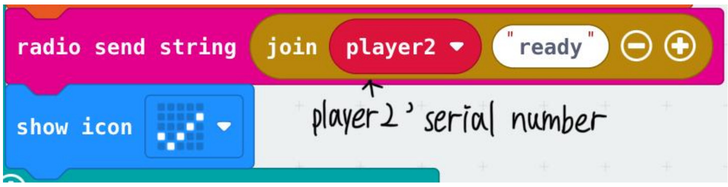
Main concept

When I saw that we need to use three microbits to act a console and two players and use two programs to play the game, I think it's not very made sense and convenient because the players rely on the console to process their uploaded data. So I delete the console in my code. Let the players exchange their data each other and process together, just like formed a virtual console. Images below.

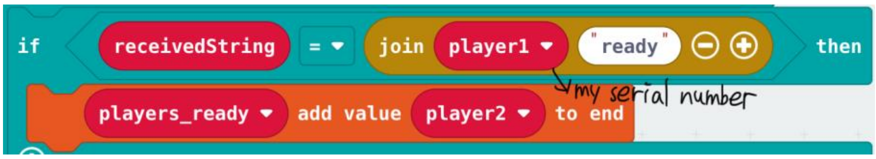

console
process
upload data
upload data
player 1
player2
players upload data to the console and process by the console only


exchange data
virtual console
process
player 1
player2
players exchange data and process together. Form a virtual console

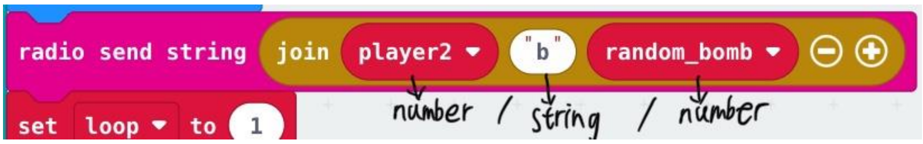
Security



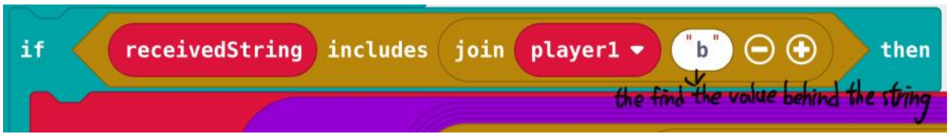
```
radio send string join player2 "ready"
show icon
```



```
if receivedString = join player1 "ready" then
    players_ready add value player2 to end
```



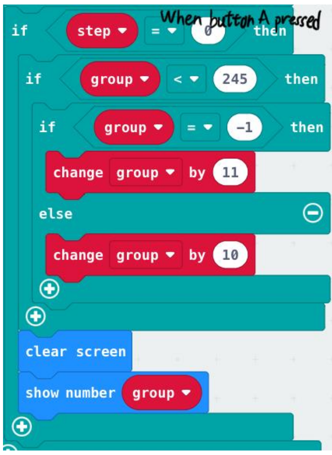
```
radio send string join player2 "b" random_bomb
set loop to 1
```



```
if receivedString includes join player1 "b" then
```

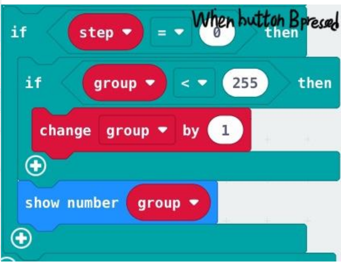
When the players are exchanging data, the data will mix with the other pair of players' so I encrypt the data before send out. The principle is adding the paired players' serial number in front of the string or number that you want to send. When that player received the whole sentence, it will check is the received serial number same to its device serial number first, if they are same, it will start reading the string or number. It is also work to encrypting variables.

Pair



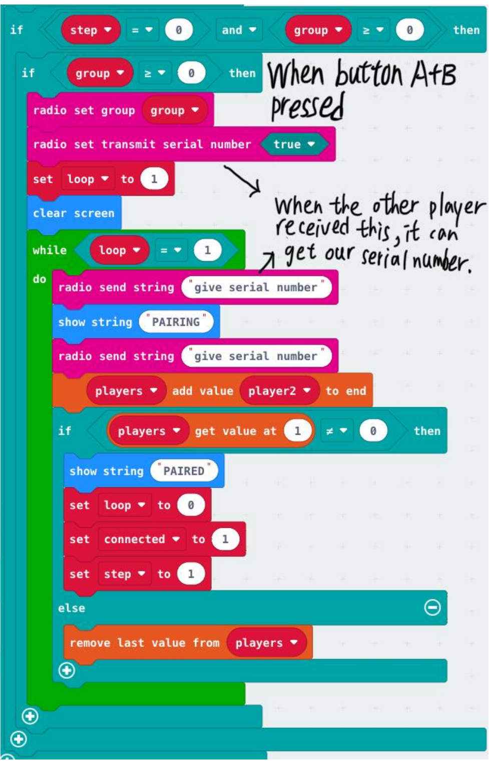
When button A pressed

```
if step = 0 then
    if group < 245 then
        if group = -1 then
            change group by 11
        else
            change group by 10
    clear screen
    show number group
```



When button B pressed

```
if step = 0 then
    if group < 255 then
        change group by 1
    show number group
```



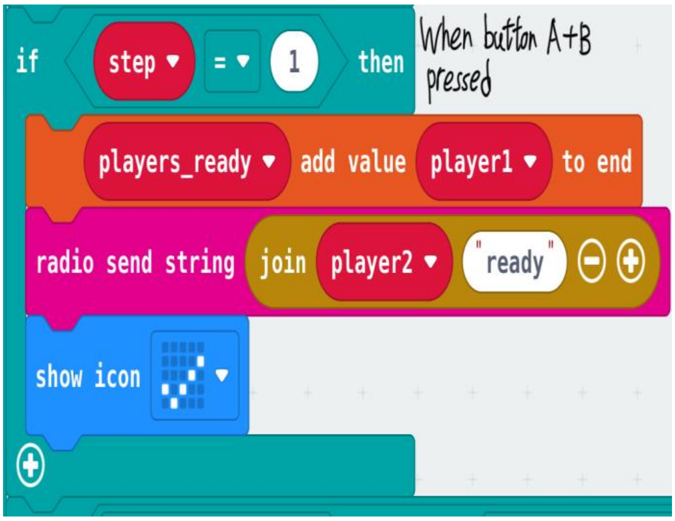
When button A+B pressed

When the other player received this, it can get our serial number.

```
if step = 0 and group ≥ 0 then
    if group ≥ 0 then
        radio set group group
        radio set transmit serial number true
        set loop to 1
        clear screen
        while loop = 1
        do
            radio send string "give serial number"
            show string "PAIRING"
            radio send string "give serial number"
            players add value player2 to end
            if players get value at 1 ≠ 0 then
                show string "PAIRED"
                set loop to 0
                set connected to 1
                set step to 1
            else
                remove last value from players
```

First, for having a higher freedom, I've made a pair system. We need to enter the radio group number(0-255) by pressing button A/B. The group number will be change by 10 when button A is pressed. When button B pressed, the radio group number will be change by 1. Then, press button A+B to start pairing.(*The entered radio group number must be same with your friend’s entered radio group number. If not, you cannot pair with your friend.)

Ready ✓

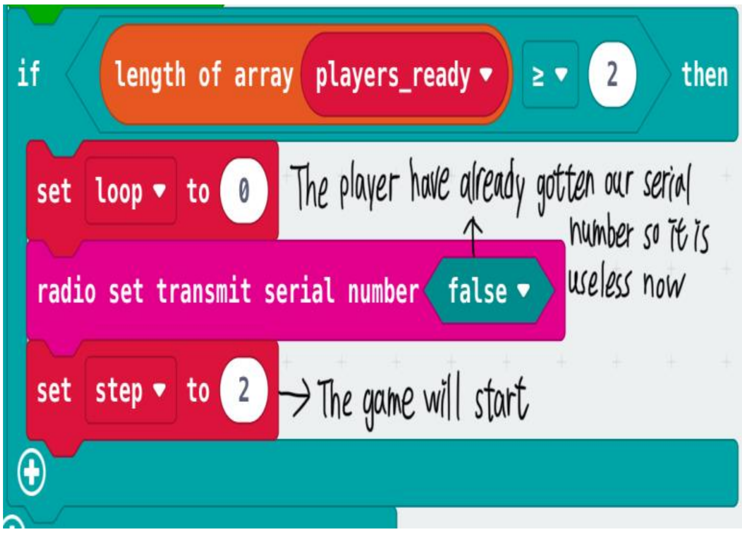


```
if step = 1 then
    players_ready add value player1 to end
    radio send string join player2 "ready"
    show icon
```

When button A+B pressed

After paired, we need to press button A+B to get ready and wait for the game start. Make sure that the game start in the same time.

When the players both got ready, the serial number will close so it cannot be gotten anymore. Then the game will start at the same time.

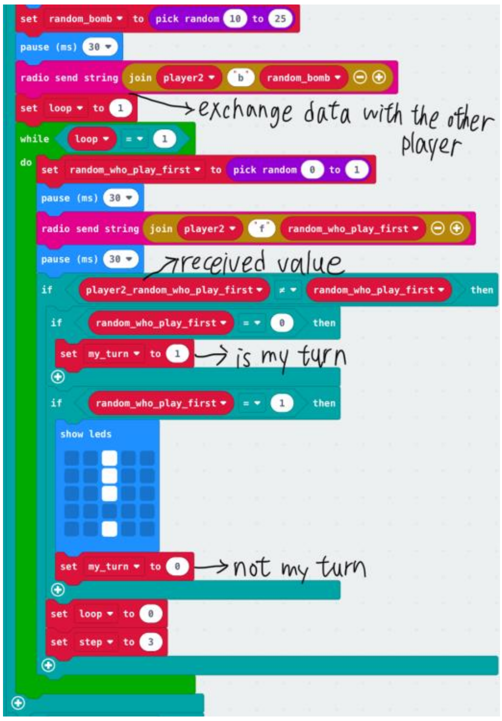


```
if length of array players_ready ≥ 2 then
    set loop to 0
    radio set transmit serial number false
    set step to 2
```

The player have already gotten our serial number so it is useless now

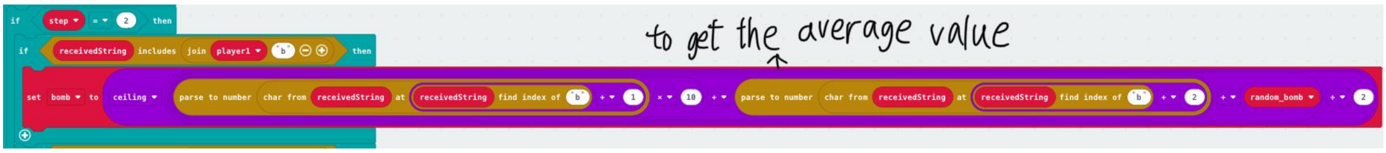
→ The game will start

Random



```
set random_bomb to pick random 10 to 25
pause (ms) 30
radio send string join player2 "b" random_bomb
set loop to 1
while loop = 1
do
    set random_who_play_first to pick random 0 to 1
    pause (ms) 30
    radio send string join player2 "f" random_who_play_first
    pause (ms) 30
    if player2_random_who_play_first ≠ random_who_play_first then
        if random_who_play_first = 0 then
            set my_turn to 1
        if random_who_play_first = 1 then
            show leds
            set my_turn to 0
        set loop to 0
        set step to 3
```

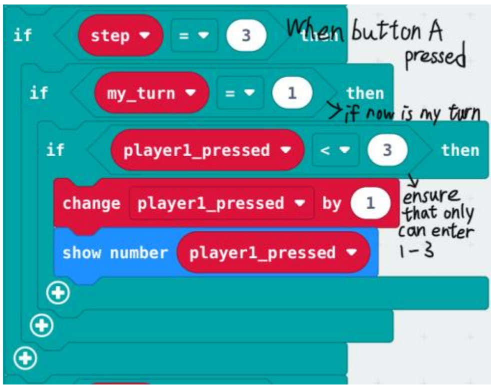
When the game started, each player will random a value of bomb and random a player to play first. Then it will send the numbers. After random a player to play first, that player will start first.



```
if step = 2 then
    if receivedString includes join player1 "b" then
        set bomb to ceiling parse to number char from receivedString at receivedString find index of "b" + 1 × 10 + parse to number char from receivedString at receivedString find index of "b" + 2 + random_bomb ÷ 2
```

Because of the random number of the players are different, so they will share their random number each other and get the average value. Make sure that their value of bomb are same. It is similar to human society, when people have different options, they will find the same point too.

Play

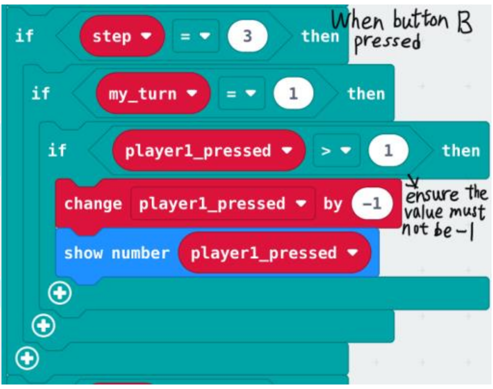


```
if step = 3 then
  if my_turn = 1 then
    if player1_pressed < 3 then
      change player1_pressed by 1
      show number player1_pressed
```

When button A pressed

if now is my turn

ensure that only can enter 1-3

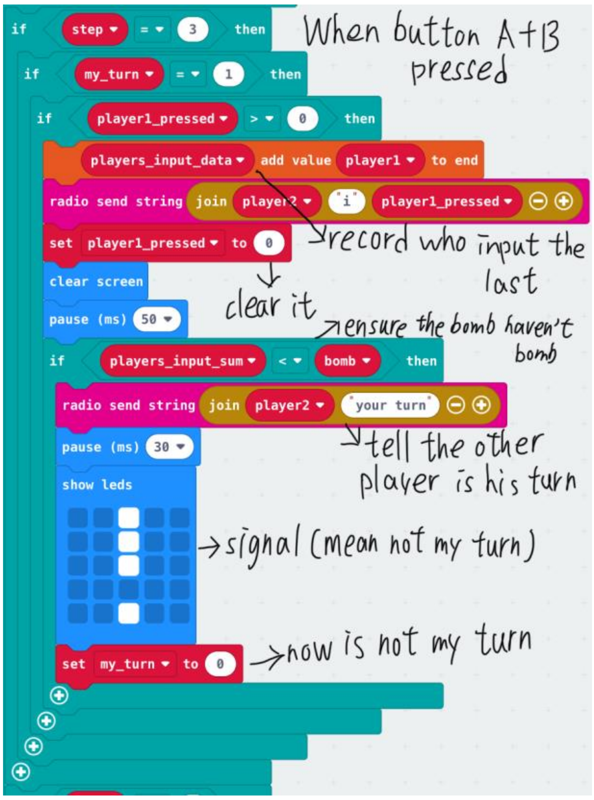


```
if step = 3 then
  if my_turn = 1 then
    if player1_pressed > 1 then
      change player1_pressed by -1
      show number player1_pressed
```

When button B pressed

ensure the value must not be -1

When it is your turn, you need to press button to increase 1 or press button B to decrease 1. (The range is 1-3)



```
if step = 3 then
  if my_turn = 1 then
    if player1_pressed > 0 then
      players_input_data add value player1 to end
      radio send string join player2 "i" player1_pressed
      set player1_pressed to 0
      clear screen
      pause (ms) 50
      if players_input_sum < bomb then
        radio send string join player2 "your turn"
        pause (ms) 30
        show leds
        set my_turn to 0
```

When button A+B pressed

record who input the last

clear it

ensure the bomb haven't bomb

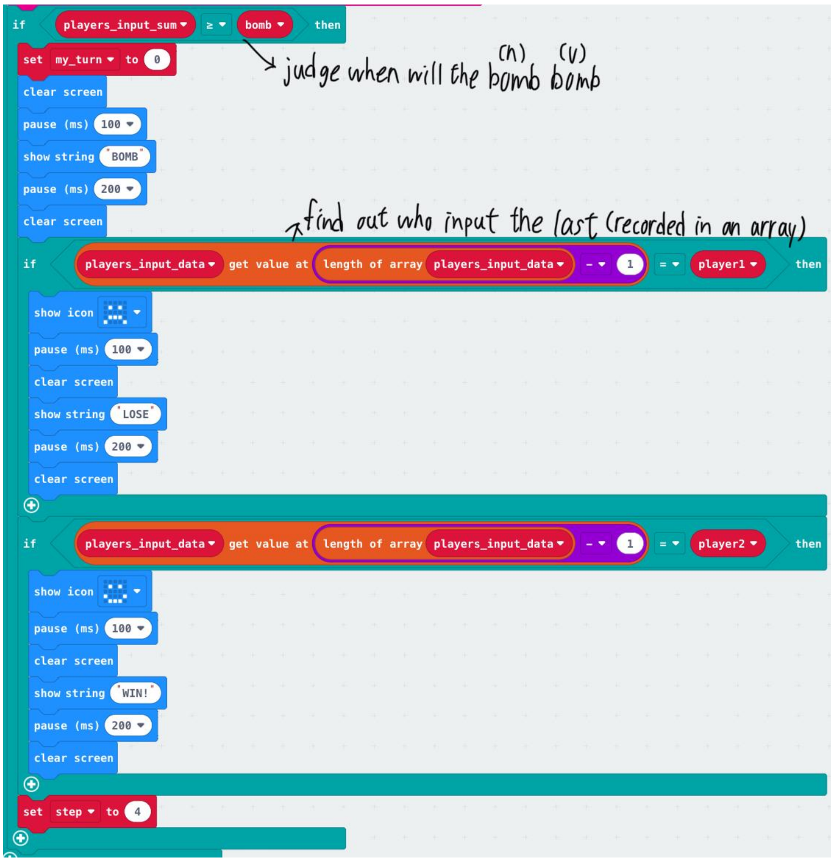
tell the other player is his turn

signal (mean not my turn)

now is not my turn

After that, press button A+B to send the value. Then it is the next player's turn.

Game over



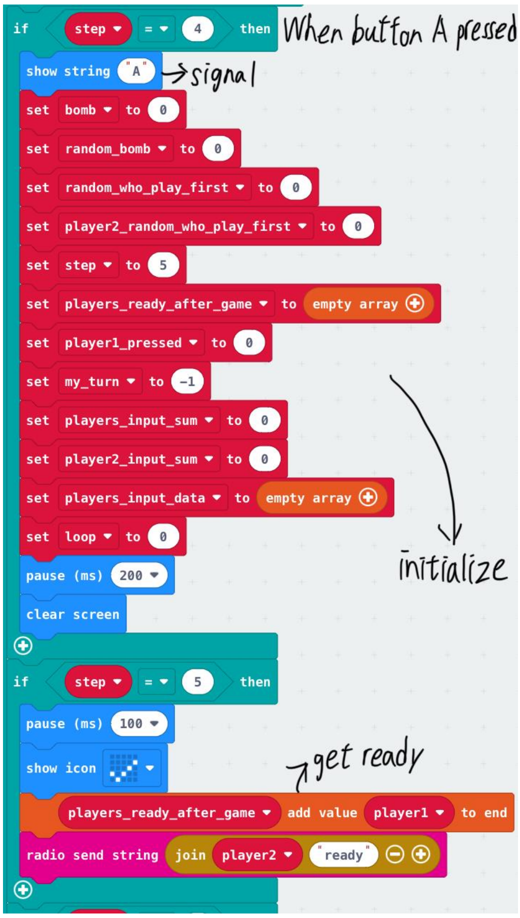
```
if players_input_sum ≥ bomb then
    set my_turn to 0
    clear screen
    pause (ms) 100
    show string "BOMB"
    pause (ms) 200
    clear screen
    if players_input_data get value at length of array players_input_data - 1 = player1 then
        show icon
        pause (ms) 100
        clear screen
        show string "LOSE"
        pause (ms) 200
        clear screen
    if players_input_data get value at length of array players_input_data - 1 = player2 then
        show icon
        pause (ms) 100
        clear screen
        show string "WIN!"
        pause (ms) 200
        clear screen
    set step to 4
```

judge when will the bomb (n) bomb (v)

find out who input the last (recorded in an array)

When the sum of the players' total pressed is higher or equal to the bomb, the game will be over and the players will find out who is the last player to input the number. If the last player is itself, it is the loser. Opposite, if the last player is not itself, it is the winner.

Rematch



```
if  step = 4  then
    show string "A"
    set bomb to 0
    set random_bomb to 0
    set random_who_play_first to 0
    set player2_random_who_play_first to 0
    set step to 5
    set players_ready_after_game to empty array
    set player1_pressed to 0
    set my_turn to -1
    set players_input_sum to 0
    set player2_input_sum to 0
    set players_input_data to empty array
    set loop to 0
    pause (ms) 200
    clear screen
if  step = 5  then
    pause (ms) 100
    show icon
    players_ready_after_game add value player1 to end
    radio send string join player2 "ready"
```

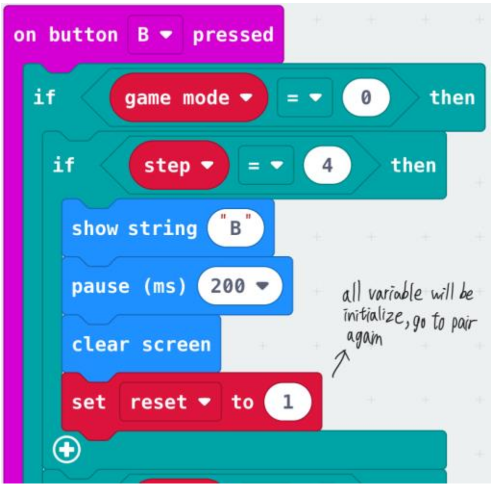
When button A pressed

→signal

initialize

get ready

After the match, if you want to have a match again with the same paired player, you no need to pair again, just press button A to get ready for the new match start. When the players both got ready, the game will start.(*There is a bug, read P.14)

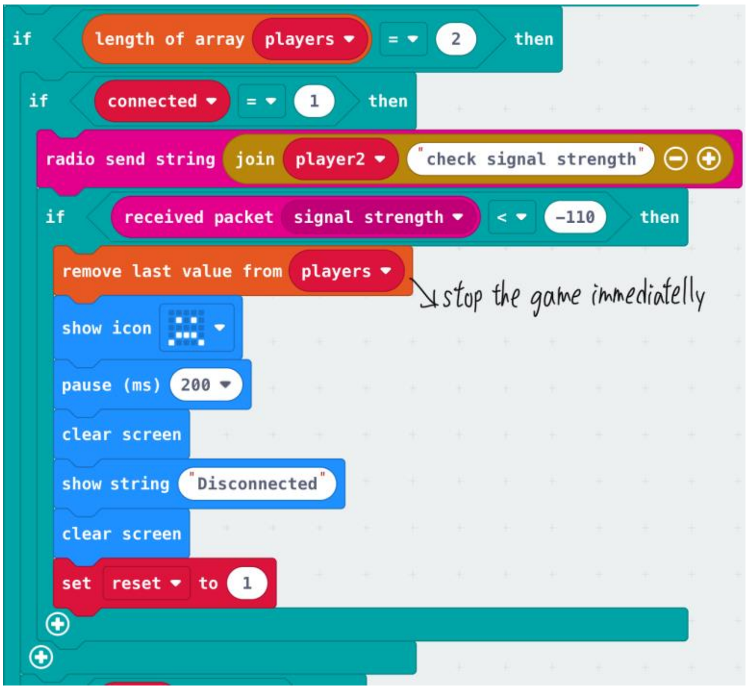


```
on button B pressed
    if  game mode = 0  then
        if  step = 4  then
            show string "B"
            pause (ms) 200
            clear screen
            set reset to 1
```

all variable will be initialize, go to pair again

Also, if you don't want, just press button B to reset the game and you can pair with other players.

Disconnect



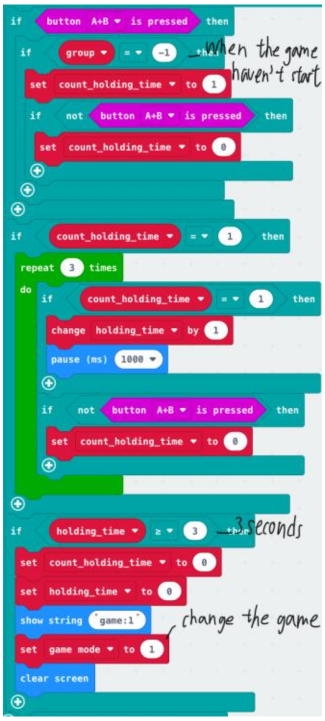
```
if length of array players = 2 then
  if connected = 1 then
    radio send string join player2 "check signal strength"
    if received packet signal strength < -110 then
      remove last value from players
      show icon
      pause (ms) 200
      clear screen
      show string "Disconnected"
      clear screen
      set reset to 1
```

stop the game immediatelly

When the distance between the players is too far, the connection of the players may be unstable. For solving it, I have limited the signal strength range. The normal range is -42 to -128, so I limited it to -42 to -110 because I thought it is a suitable and acceptable range. When the signal strength exceeded the range, the game will stop immediately and disconnect with the other player. Then the game will reset so you need to pair again, remember to make sure that the signal strength is in the range otherwise it cannot pair successfully.

Hidden game ◡̈

Now the microbit have a multiplayer game, but when there is only one player, the game cannot work, so I added stand-alone game to the program, “catch the coin”. After you pressed button A+B to start this game, the top of the screen will drop a coin that’s drop from a random place. Then you need to control the sprite by button A/B to catch the dropping coin at the bottom. If you catch the coin successfully, the score will change by 1 and the other one coin will be dropped again. If you couldn’t catch the coin, the game will end and show the score.



```
if button A+B is pressed then
    if group = -1 then
        set count_holding_time to 1
        if not button A+B is pressed then
            set count_holding_time to 0
if count_holding_time = 1 then
    repeat 3 times
    do
        if count_holding_time = 1 then
            change holding_time by 1
            pause (ms) 1000
        if not button A+B is pressed then
            set count_holding_time to 0
if holding_time ≥ 3 then
    set count_holding_time to 0
    set holding_time to 0
    show string "game:1"
    set game mode to 1
    clear screen
```

when the game haven't start

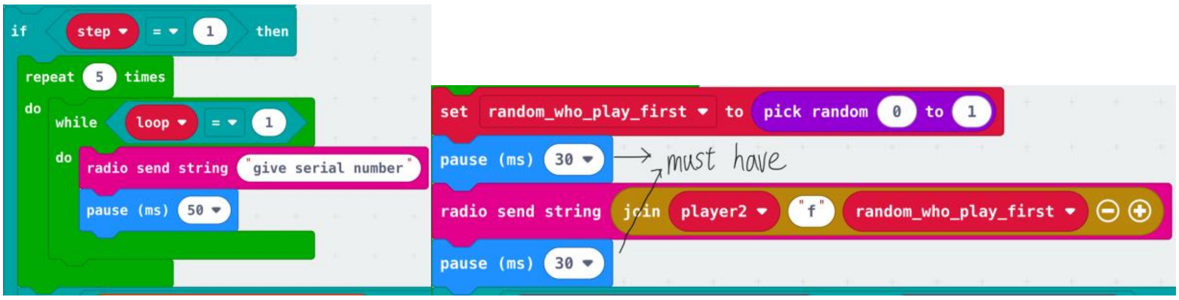
3 seconds

change the game

Then how to change the game to the stand-alone game form the multiplayer game? Before you enter the radio group number, hold button A+B three seconds to change the game. You can change the game back to the multiplayer game by the same way.

Bugs

In programming, almost each program have bugs, in this program, there are many bugs also. So how did I fix these bug? Just try! I thought a lot of different solutions to solve the bugs. Although the solutions look so strange and not make sence, they worked! You can see many strange places in my code, All of them are for fixing bugs. Example:



```
if  step = 1  then
  repeat 5 times
  do
    while  loop = 1
    do
      radio send string "give serial number"
      pause (ms) 50
```

```
set random_who_play_first to pick random 0 to 1
pause (ms) 30
radio send string join player2 "f" random_who_play_first
pause (ms) 30
```

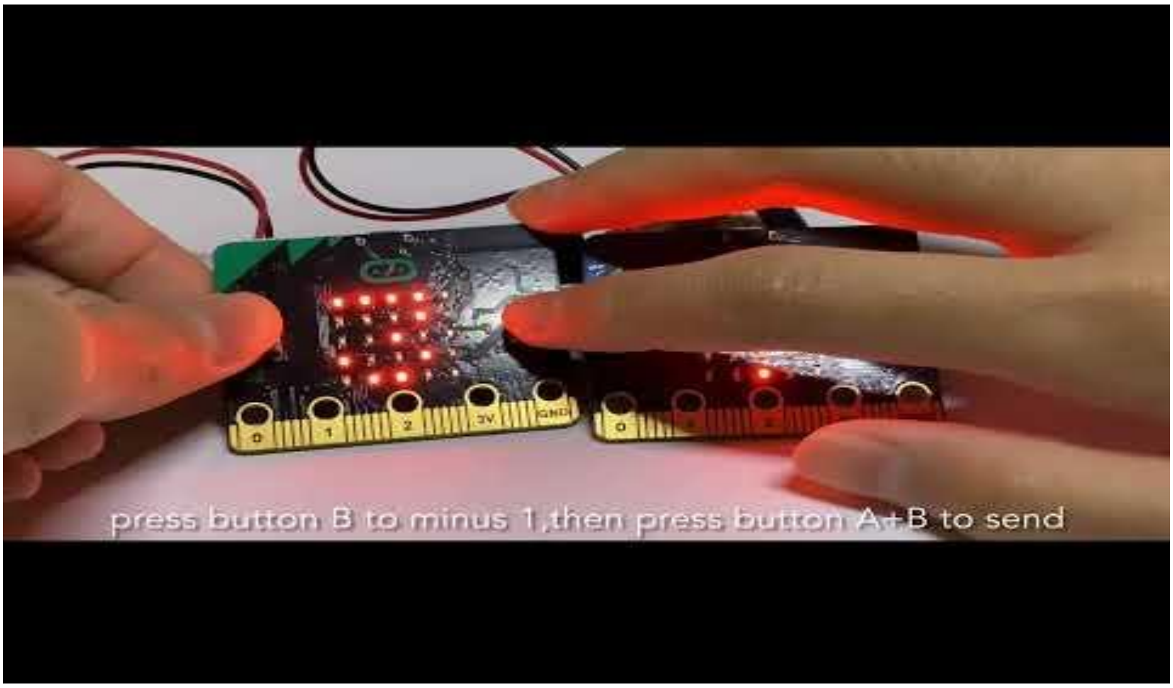
must have

But there is one bug haven’t fixed yet. It is in the rematch part. If the players want to start a new match, the players must press button A at the same time. Otherwise one player will start the game, one player won’t.

Resources

Running video

https://youtu.be/7y3eGbMaajA



Program file & Others

https://github.com/kakaho0902/Bomberman-Game/tree/main

Ending 👋

Although I started work on this project since 28th of November, I handed in this project late. I’m sorry about that. This project is quite challenging, I have learnt a lot. That’s the end, thank you for your patience!

*A thank list in the next page.

Thank list ♡

- **Mr. Fan**

(He taught me new things example radio, array, serial number, signal strength.)

- **Wong Yin Yan (2D)**
- **Wong Chun Hei (2C)**
- **Ho Hing Kiu (2D)**
- **Lee Man Ho (2C)**
- **Lam Ka Pui (2D)**

(They borrowed their microbit to me for testing the program.)

- **Ho Hing Kiu (2D)**
- **Lam Ka Pui (2D)**

(They helped me to test my program.)

- **Ms. Chu**

(She praised me, made me became more confident.)

Thank for all of you!

-The End-