**C Project Report: Super Mario**

**Abstract**

Do you have a childhood memory of a bouncing, mushroom-topped figure? Starting with the red and white machine, Mario became a childhood playmate for many people, carrying their childhood memories.

We want to design a game using all the knowledge we have learned before. Super Mario is chosen as the theme because our group are all the players of the game and it also carries our precious memories.

In our game programming, we met some central problems.

**1. Introduction & Problem Statement**

In our childhood, games may be one of the best friends of us. Super Mario is a pretty classical one and it’s just like an old friend of us. As a result, after we have learned Language C, we decided to try our best to remake this game with the knowledge we have learned.

As beginners, we also meet some problems. The first one is interface. This is really important. However, most of this kind of game is built by Python or Java, seldom in C. Consequently, it’s really a challenge for us because there’s not so much proper code for us to consult. After preparing and referring to some datum, we decided to use ACLLib, a function library based on WIN32API, which provides a relatively simple way to do Windows programs.

What’s more, the movement control is also a big problem. On account of there is little source code, we need to think of the algorithm of movement by ourselves. Of course, the final aim of our project is to complete the whole game and pass the happiness to game players.

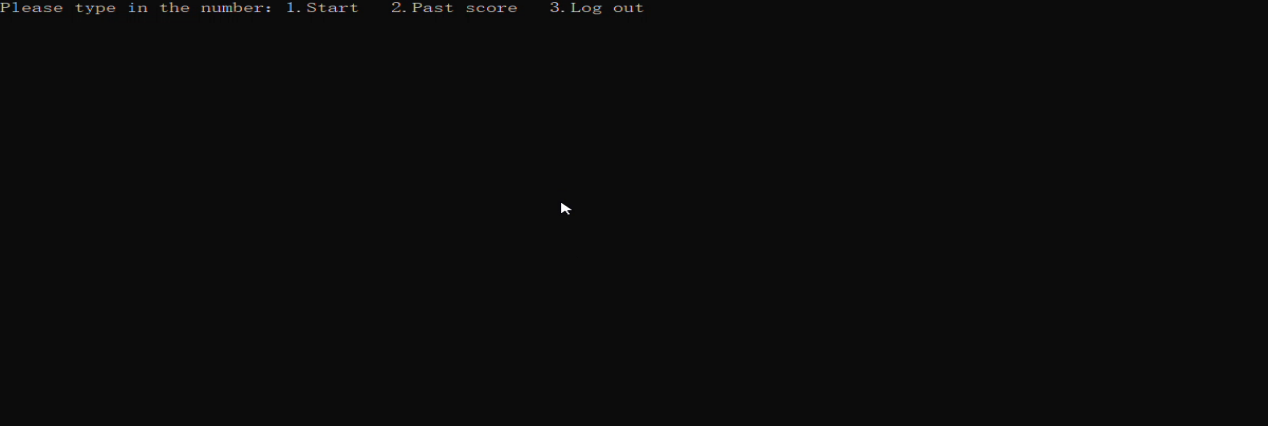
**2.Analysis**

To make a relatively complete game, we think that the final program must have a beautiful game interface, which the users can have buttons such as registration, login, and exit; flexible operation and control so that the Mario could move in a certain speed and players just need to press up, down, left, right to change the moving direction; the score judgement of the game, for instance, if the NPC catches Mario, the HP of Mario will minus one and the game will terminate when HP comes to zero or when the Mario gets enough coins to let him win the game.

Now that we have decided the main direction, we need to try to understand the problems on hand and to be solved.

**2.1 Build a game interface**

For the new players, they can create their own account by choosing the registration button and then log in to start the game; for the old players, they can log in and start the game directly. Besides, they also can see their past scores in order if they want to.



**2.2 The movement of Mario**

As the basic part of this game, the movement of the Mario and the NPC can be said the most important. How to make these characters move together and fluently is the most complex part of our program because the program is single thread.

What’s more, the movement of the NPC cannot be random because it may turn around at one place all the time. As a result, we decide to make the judgement at the crossing. Then how to judge a crossing? That’s quite a problem to us.

In addition, the movement of the Mario is controlled by our input commands. We need to press up, down, left, and right to change the direction. But we can’t make the Mario hit on the wall or NPC. We need it follows the player’s commands then eat enough bills.



**2.3 The judgement of winning and death**

When Mario gets enough coins and it will enter the next level but if it loses all his hearts, it will lose the game.

**2.4 The switch between the levels**

To meet the need of different players, we design two levels and in the second level, the NPC will be faster so the it’s more dangerous. The players can find the fun of the game.

LEVEL 1



LEVEL 2

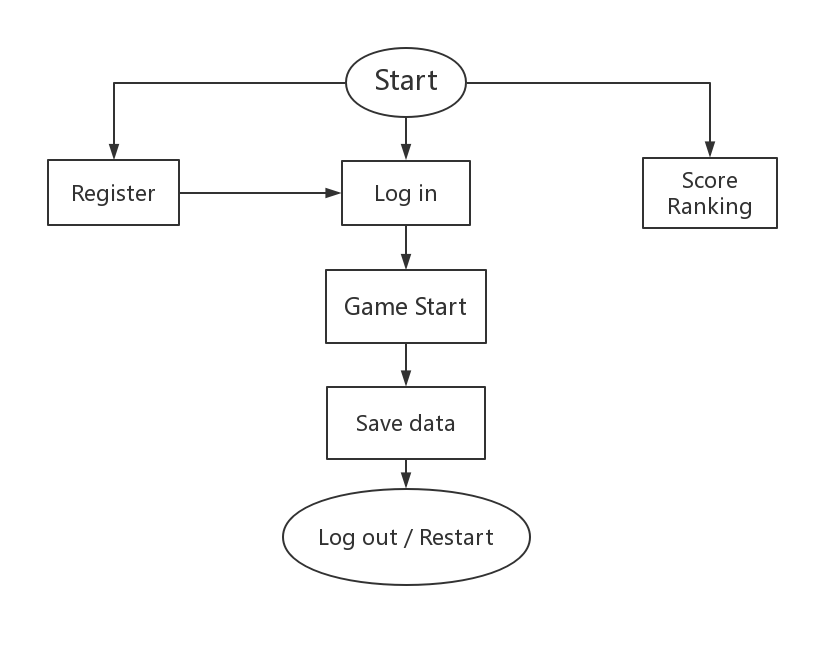


## 

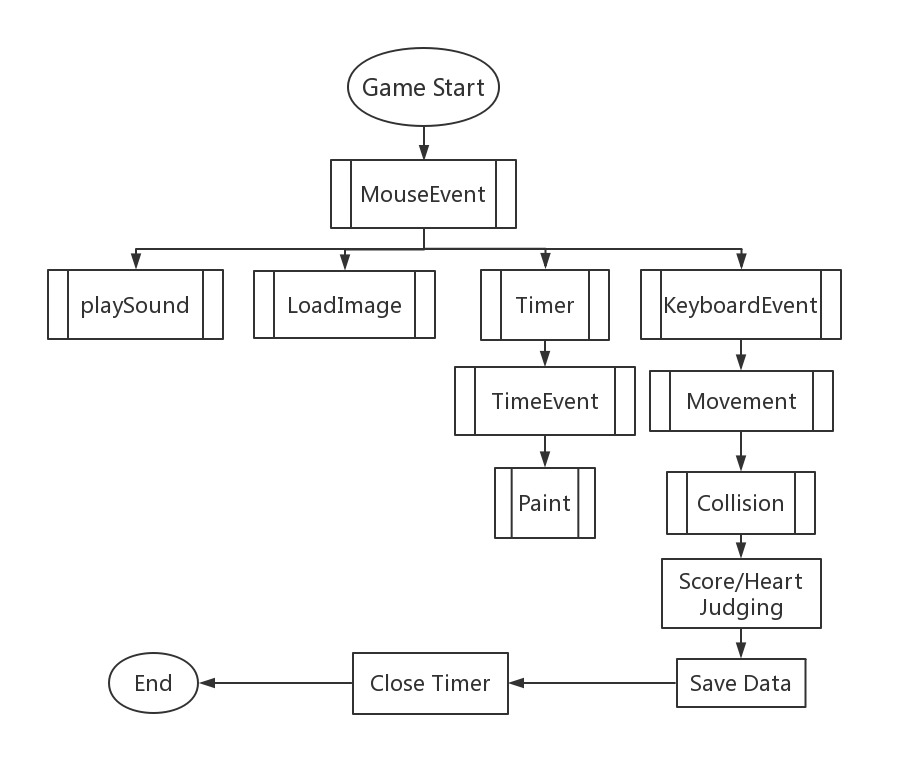
**3.Design**

* 1. **The overall framework**

After analysis, we first designed the overall framework of the program.



After running, there will be a start interface with three different choices: register for new players, log in for old players, and view leader-board. If the player chooses to view the leader-board, the highest six scores and player records will appear. During the registration process, if the user name players entered is already occupied, they will be prompted to change the user name. After successful registration, players can log in and enter the game. If the player has already had an account, they can enter the game directly. After starting and completing a game, the data is stored directly in the history file. The player is then presented with a restart/log out interface and can choose from the options as appropriate.



In actual game progress, mouse-controlled functions are first run. Players can then use the mouse to start the game. After the player clicks the start button, background music playing, images loading, and keyboard control functions will be triggered, and the Timer will be started. Functions for time event will run whenever it reaches the preset time. By doing so, all the images can be painted based on timing results. Since we have already call the keyboard control function, now players can use keyboard to let the character move. During the game, collision function is utilized for score and heart judging. When the game is over, the data will be saved in history file and the Timer will be closed too. Then, it will switch to Restart/ Log out interface.

**4.Implementation**

**4.1 ACLLIB and C**

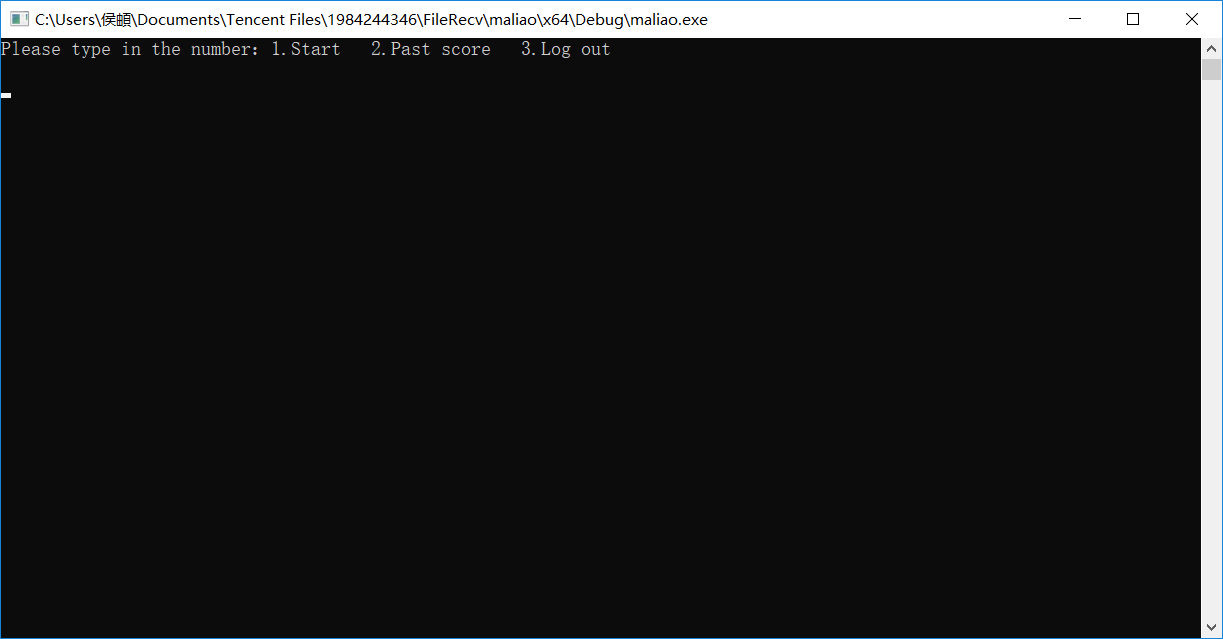
In terms of a Mario game, we tried to make all the images move and paint all the images by ourselves. According to a short research on the website, we noticed that most of the C programming games are made by EasyX. However, we had problems using ‘ scanf ’ function. So finally we chose to use ACLLIB library to make our dreams come true. Also we decided to use C language to write down our codes.

We need a head and a resource documents to do that.



**4.2 Interface**

When it comes to Interface, we wondered for a long time to design a beautiful and useful one. Unfortunately, the Acllib we used isn’t supposed for the ‘scanf ’ and ‘printf ’ functions. So we finally chose to finish it by command-line window in order to print what we want to show to players. the following code is essential.



**CODE:**

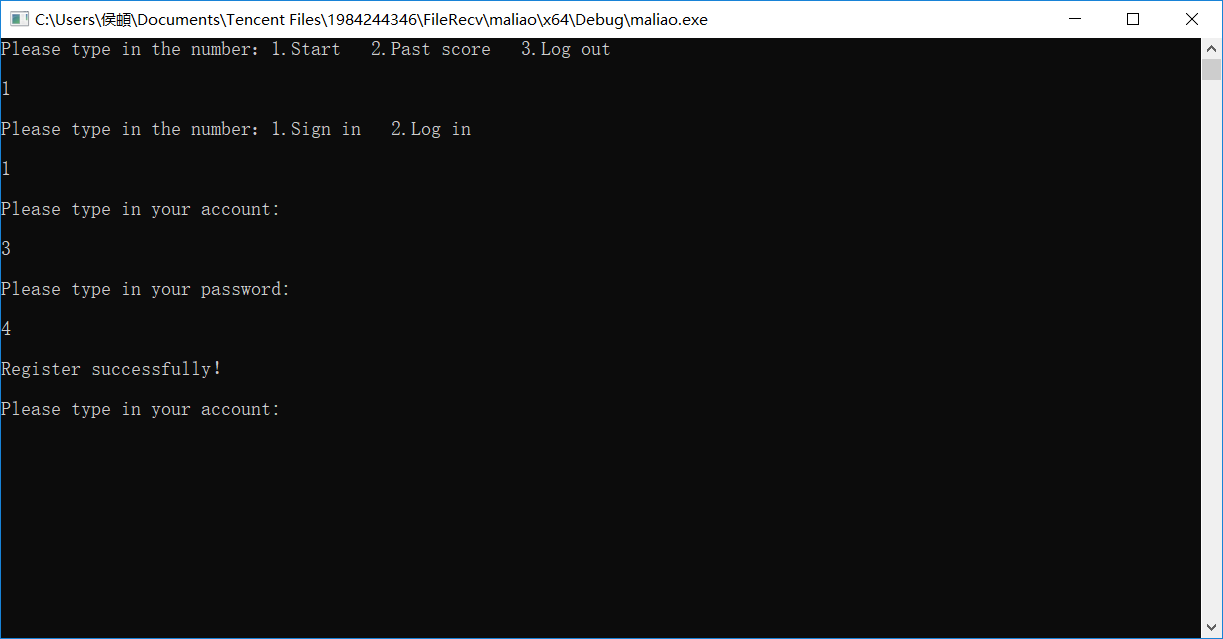
int x, y;

initConsole();

**4.3 Login and Register**

We wanted to design the Login and Resister functions to record the information of our players. To achieve this goal, we insert two text documents to record all of them. The players only need to choose one or two to start the function.

In fact, we also need to deal with so many possible problems such as unrecognized input or wrong information being typed. So after finishing the protocol of our function, we invited different testers to login in the game so that we can find out the bugs hiding in the code. Finally, we figured out all the solutions to the problem.



**CODE:**

printf("Please type in the number：1.Start 2.Past score 3.Log out\n\n");

scanf\_s("%d",&x);

printf\_s("\n");

if (x == 1)

{

printf("Please type in the number：1.Sign in 2.Log in\n\n");

scanf\_s("%d", &y);

printf\_s("\n");

if (y == 1)

{

registers();

Login();

}

else if (y == 2) {

Login();

}

}

CODE for registers.c:

#include"head.h"

int registers()//注册账号密码

{

fopen\_s(&fp, "Users1.txt", "r");

fscanf\_s(fp, "%s%s", b.id, sizeof(b.id), b.pwd, sizeof(b.pwd));

//InputBox(a.id, 11, "请输入账号");

printf\_s("Please type in your account:\n\n");

scanf\_s("%d", a.id);

printf\_s("\n");

while (1)

{

if (strcmp(a.id, b.id) != 0)//如果两串字符串不相等

{

if (!feof(fp))//如果未至文件末尾

{

fscanf\_s(fp, "%s%s", b.id, sizeof(b.id), b.pwd, sizeof(b.pwd));

}

else break;

}

else

{

printf("This account has already been registered\n\n");

fclose(fp);

\_getch();

exit(0);

}

}

fclose(fp);

printf\_s("Please type in your password:\n\n");

scanf\_s("%d", a.pwd);

printf\_s("\n");

fopen\_s(&fp, "Users1.txt", "a");

fprintf\_s(fp, "%s %s", a.id, a.pwd);

printf("Register successfully！\n\n");

fclose(fp);

return 0;

}

CODE FOR Login.c:

#include"head.h"

int Login()//登陆账号密码

{

fopen\_s(&fp, "Users1.txt", "r");

fscanf\_s(fp, "%s%s", b.id, sizeof(b.id), b.pwd, sizeof(b.pwd));

printf\_s("Please type in your account:\n\n");

scanf\_s("%d",a.id);

printf\_s("\n");

while (1)

{

if (strcmp(a.id, b.id) == 0) break;//如果找到了这个用户名

else

{

if (!feof(fp))//如果文件未读完

fscanf\_s(fp, "%s%s", b.id, sizeof(b.id), b.pwd, sizeof(b.pwd));

else

{

printf\_s("This account doesn't exist!\n\n");

fclose(fp);

\_getch();

exit(0);

}

}

}

printf\_s("Please type in your password:\n\n");

scanf\_s("%d",a.pwd);

printf\_s("\n");

if (strcmp(a.pwd, b.pwd) == 0)//如果密码匹配

{

fclose(fp);

printf\_s("Login is successful\n\n");

//initgraph(640, 480);

}

else

{

printf\_s( "This password is not correct\n\n");

\_getch();

exit(0);

}

return 0;

}

**4.4 Map Drawing**

In terms of how to implement the mapping, we searched the Internet for a lot of information and finally decided to draw it with ACLLIB. In order to be able to initialize the graphical interface, the following code is essential.



**CODE:**

initWindow("super\_Mario", DEFAULT, DEFAULT, winWidth, winHeight);

imageLoading();

Then let me show a small part of the code of our map.

beginPaint();

putImageScale(&background\_2\_img, background\_2\_x, background\_2\_y, background\_2\_width, background\_2\_height);

putImageTransparent(&start\_img, 275, 100, 150, 60, BLACK);

endPaint();

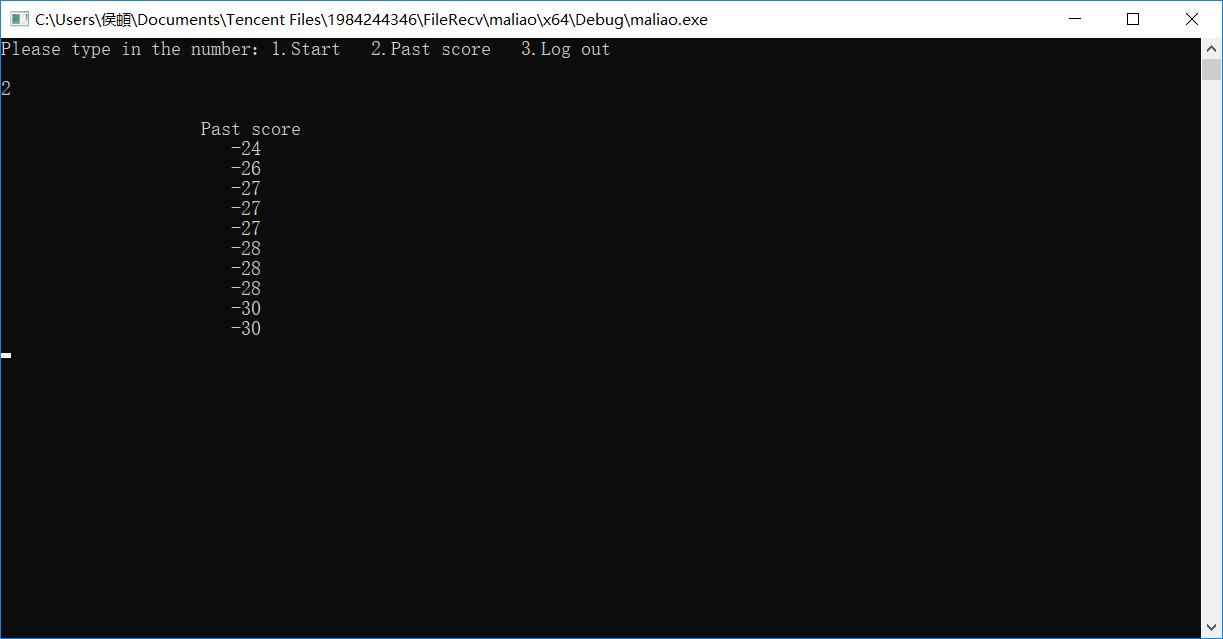
registerKeyboardEvent(keyEvent);

registerTimerEvent(timerEvent);

return 0;

**4.5 Past Score**

If we can design a function to record all the past scores, the game will be more challenging and great. We did a research on the website and found a practical merge-sort way to achieve our goal.



**CODE:**

// merge sort

#include "head.h"

#define NUM 10

int merge(int arr[], int start, int middle, int end)

{

int n1 = middle - start + 1;

int n2 = end - middle;

int i, j, k;

int \*L = (int\*)malloc((n1 + 1) \* sizeof(int));

int \*R = (int\*)malloc((n2 + 1) \* sizeof(int));

for (i = 0; i < n1; i++)

{

L[i] = arr[start + i];

}

for (j = 0; j < n2; j++)

{

R[j] = arr[middle + j + 1];

}

L[n1] = INT\_MAX;

R[n2] = INT\_MAX;

i = 0;

j = 0;

for (k = start; k <= end; k++)

{

if (L[i] <= R[j])

{

arr[k] = L[i];

i++;

}

else

{

arr[k] = R[j];

j++;

}

}

free(L);

free(R);

L = NULL;

R = NULL;

return 0;

}

int mergeSort(int arr[], int start, int end)

{

int middle = (start + end) / 2;

if (start < end)

{

mergeSort(arr, start, middle);

mergeSort(arr, middle + 1, end);

merge(arr, start, middle, end);

}

return 0;

}

int scoreSort()

{

int y[NUM];

fopen\_s(&fp1, "score.txt", "r");

for (int x = 0; x < NUM; x++)

fscanf\_s(fp1, "%d", &y[x]);

mergeSort(y, 0, NUM - 1);

fclose(fp1);

for (int x = NUM-1; x >= 0; x--)

printf(" %d\n", y[x]);

return 0;

}

**4.6 Drawing image(如何实现非矩形图及背景透明的具体尝试过程)**

We tried a lot to make the background transparent, until we suddenly found a function in ACLLib that made it transparent for certain colors. Thus, the problem is easily transformed and solved smoothly.

**Code:**

putImageTransparent(&restart\_img, 275, 100, 150, 60, BLACK);

**4.7 Judging collision**

The determination of score and loss is actually a collision and a two-way collision. It is achieved by judging the overlap of images. In the specific implementation process, we divided the collision into four types for judgment: left bottom, left top, right bottom, right top.

**Code:**

{

rectmarie.x = marie\_x;

rectmarie.y = marie\_y;

rectmarie.w = marie\_width;

rectmarie.h = marie\_height;

for (int i = 0; i < 1; i++)

{

if (goodmushsprite[i])// 不为空

{

rectgood\_mush[i].x = goodmushsprite[i]->x;

rectgood\_mush[i].y = goodmushsprite[i]->y;

rectgood\_mush[i].w = goodmushsprite[i]->width;

rectgood\_mush[i].h = goodmushsprite[i]->height;

if (collision(rectmarie, rectgood\_mush[i]) == 1)

{

score += 10;

heart += 3;

goodmushsprite[i] = NULL;

}

}

}

for (int i = 0; i < numbill; i++)

{

if (billsprite1[i])// 不为空

{

rectbill[i].x = billsprite1[i]->x;

rectbill[i].y = billsprite1[i]->y;

rectbill[i].w = billsprite1[i]->width;

rectbill[i].h = billsprite1[i]->height;

if (collision(rectmarie, rectbill[i]) == 1)

{

score += 2;

billsprite1[i] = NULL;

}

}

}

for (int i = 0; i < numcoin1; i++)

{

if (coinsprite1[i])// 不为空

{

rectcoin1[i].x = coinsprite1[i]->x;

rectcoin1[i].y = coinsprite1[i]->y;

rectcoin1[i].w = coinsprite1[i]->width;

rectcoin1[i].h = coinsprite1[i]->height;

if (collision(rectmarie, rectcoin1[i]) == 1)

{

score += 3;

coinsprite1[i] = NULL;

}

}

}

for (int i = 0; i < numcoin2; i++)

{

if (coinsprite2[i])// 不为空

{

rectcoin2[i].x = coinsprite2[i]->x;

rectcoin2[i].y = coinsprite2[i]->y;

rectcoin2[i].w = coinsprite2[i]->width;

rectcoin2[i].h = coinsprite2[i]->height;

if (collision(rectmarie, rectcoin2[i]) == 1)

{

score -= 1;

coinsprite2[i] = NULL;

}

}

}

}

rectbad\_mask.x = bad\_mask\_x;

rectbad\_mask.y = bad\_mask\_y;

rectbad\_mask.h = bad\_mask\_height;

rectbad\_mask.w = bad\_mask\_width;

if (collision(rectmarie, rectbad\_mask) == 1)

{

score -= 5;

heart -= 1;

marie\_x += 10;

}

rectgreen\_turtle.x = bad\_green\_turtle\_x;

rectgreen\_turtle.y = bad\_green\_turtle\_y;

rectgreen\_turtle.h = bad\_green\_turtle\_height;

rectgreen\_turtle.w = bad\_green\_turtle\_width;

if (collision(rectmarie, rectgreen\_turtle) == 1)

{

score -= 3;

heart -= 1;

marie\_x += 10;

}

rectbad\_missile.x = bad\_missile\_x;

rectbad\_missile.y = bad\_missile\_y;

rectbad\_missile.h = bad\_missile\_height;

rectbad\_missile.w = bad\_missile\_width;

if (collision(rectmarie, rectbad\_missile) == 1)

{

score -= 2;

heart -= 1;

marie\_x += 10;

}

rectbad\_mush.x = bad\_mush\_x;

rectbad\_mush.y = bad\_mush\_y;

rectbad\_mush.h = bad\_mush\_height;

rectbad\_mush.w = bad\_mush\_width;

if (collision(rectmarie, rectbad\_mush) == 1)

{

score -= 10;

heart -= 3;

marie\_x += 10;

}

rectbad\_snake.x = bad\_snake\_x;

rectbad\_snake.y = bad\_snake\_y;

rectbad\_snake.h = bad\_snake\_height;

rectbad\_snake.w = bad\_snake\_width;

if (collision(rectmarie, rectbad\_snake) == 1)

{

score -= 5;

heart -= 1;

marie\_x += 10;

}

paint();//必须重新绘制

return 0;

}

int timerEvent(int id)//当500ms到了会发生什么

{

int i;

if (id == 0)

{

bad\_missile\_x -= dx\_bad\_missile;

//paint();

timecounter++;

if (timecounter % 100 == 0)

bad\_missile\_x = 930;

rectbad\_missile.x = bad\_missile\_x;

rectbad\_missile.y = bad\_missile\_y;

rectbad\_missile.h = bad\_missile\_height;

rectbad\_missile.w = bad\_missile\_width;

if (collision(rectbad\_missile, rectmarie) == 1)

{

score -= 2;

heart -= 1;

marie\_x += 10;

}

}

else if (id == 1 && score >= 0)

{

//if (point <= 50) return;

bad\_mask\_x += dx\_bad\_mask;

if (bad\_mask\_x <= 0 || bad\_mask\_x + bad\_mask\_width >= winWidth)

dx\_bad\_mask = -dx\_bad\_mask;

//paint();

rectbad\_mask.x = bad\_mask\_x;

rectbad\_mask.y = bad\_mask\_y;

rectbad\_mask.h = bad\_mask\_height;

rectbad\_mask.w = bad\_mask\_width;

if (collision(rectbad\_mask, rectmarie) == 1)

{

score -= 5;

heart -= 1;

marie\_x += 10;

}

}

else if (id == 2)

{

//int i = 0;

for (i = 0; i <= numbill; i++)//bill将所有出现的bill进行移动

{

if (billsprite1[i])

{

billsprite1[i]->x += billsprite1[i]->dx;

billsprite1[i]->y += billsprite1[i]->dy;

if (billsprite1[i]->x <= 0 || billsprite1[i]->x + billsprite1[i]->width >= winWidth)

billsprite1[i]->dx = -billsprite1[i]->dx;

if (billsprite1[i]->y <= 0 || billsprite1[i]->y + billsprite1[i]->height + wall\_2\_height >= winHeight)

billsprite1[i]->dy = -billsprite1[i]->dy;

rectmarie.x = marie\_x;

rectmarie.y = marie\_y;

rectmarie.w = marie\_width;

rectmarie.h = marie\_height;

rectbill[i].x = billsprite1[i]->x;

rectbill[i].y = billsprite1[i]->y;

rectbill[i].w = billsprite1[i]->width;

rectbill[i].h = billsprite1[i]->height;

if (collision(rectbill[i], rectmarie) == 1)

{

score += 2;

billsprite1[i] = NULL;

}

}

}

}

else if (id == 3)

{

if (numbill < maxbill)

numbill++;

else

numbill = 0;

if (billsprite1[numbill] == NULL)

{

billsprite1[numbill] = (sprite\*)malloc(sizeof(sprite));

billsprite1[numbill]->width = 20;

billsprite1[numbill]->height = 20;

billsprite1[numbill]->x = rand() % (winWidth - billsprite1[numbill]->width);

billsprite1[numbill]->y = rand() % (winHeight - wall\_2\_height - billsprite1[numbill]->height);

billsprite1[numbill]->dx = 2;

billsprite1[numbill]->dy = 2;

}

}

else if (id == 4)

{

bad\_green\_turtle\_x += dx\_bad\_green\_turtle;

if (bad\_green\_turtle\_x >= 400 || bad\_green\_turtle\_x <= 0)

dx\_bad\_green\_turtle \*= -1;

rectgreen\_turtle.x = bad\_green\_turtle\_x;

rectgreen\_turtle.y = bad\_green\_turtle\_y;

rectgreen\_turtle.h = bad\_green\_turtle\_height;

rectgreen\_turtle.w = bad\_green\_turtle\_width;

if (collision(rectgreen\_turtle, rectmarie) == 1)

{

score -= 3;

heart -= 1;

marie\_x += 10;

}

}

else if (id == 5)

{

for (int i = 0; i < 1; i++)

{

if (goodmushsprite[i])

{

goodmushsprite[i]->x += goodmushsprite[i]->dx;

if (goodmushsprite[i]->x >= 400 || goodmushsprite[i]->x <= 0)

goodmushsprite[i]->dx \*= -1;

rectmarie.x = marie\_x;

rectmarie.y = marie\_y;

rectmarie.w = marie\_width;

rectmarie.h = marie\_height;

rectgood\_mush[i].x = goodmushsprite[i]->x;

rectgood\_mush[i].y = goodmushsprite[i]->y;

rectgood\_mush[i].w = goodmushsprite[i]->width;

rectgood\_mush[i].h = goodmushsprite[i]->height;

if (collision(rectgood\_mush[i], rectmarie) == 1)

{

score += 10;

heart += 3;

goodmushsprite[i] = NULL;

}

}

}

}

//coin

else if (id == 7)

{

//int i = 0;

for (i = 0; i <= numcoin1; i++)//coin将所有出现的coin进行移动

{

if (coinsprite1[i])

{

coinsprite1[i]->x += coinsprite1[i]->dx;

coinsprite1[i]->y += coinsprite1[i]->dy;

if (coinsprite1[i]->x <= 0 || coinsprite1[i]->x + coinsprite1[i]->width >= winWidth)

coinsprite1[i]->dx = -coinsprite1[i]->dx;

if (coinsprite1[i]->y <= 0 || coinsprite1[i]->y + coinsprite1[i]->height + wall\_2\_height >= winHeight)

coinsprite1[i]->dy = -coinsprite1[i]->dy;

rectmarie.x = marie\_x;

rectmarie.y = marie\_y;

rectmarie.w = marie\_width;

rectmarie.h = marie\_height;

rectcoin1[i].x = coinsprite1[i]->x;

rectcoin1[i].y = coinsprite1[i]->y;

rectcoin1[i].w = coinsprite1[i]->width;

rectcoin1[i].h = coinsprite1[i]->height;

if (collision(rectcoin1[i], rectmarie) == 1)

{

score += 3;

coinsprite1[i] = NULL;

}

}

}

}

else if (id == 8)

{

if (numcoin1 < maxcoin1)

numcoin1++;

else

numcoin1 = 0;

if (coinsprite1[numcoin1] == NULL)

{

coinsprite1[numcoin1] = (sprite\*)malloc(sizeof(sprite));

coinsprite1[numcoin1]->width = 20;

coinsprite1[numcoin1]->height = 20;

coinsprite1[numcoin1]->x = rand() % (winWidth - coinsprite1[numcoin1]->width);

coinsprite1[numcoin1]->y = rand() % (winHeight - wall\_2\_height - coinsprite1[numcoin1]->height);

coinsprite1[numcoin1]->dx = 0;

coinsprite1[numcoin1]->dy = 3;

}

}

//

else if (id == 9)

{

//int i = 0;

for (i = 0; i <= numcoin2; i++)//coin将所有出现的coin进行移动

{

if (coinsprite2[i])

{

coinsprite2[i]->x += coinsprite2[i]->dx;

coinsprite2[i]->y += coinsprite2[i]->dy;

if (coinsprite2[i]->x <= 0 || coinsprite2[i]->x + coinsprite2[i]->width >= winWidth)

coinsprite2[i]->dx = -coinsprite2[i]->dx;

if (coinsprite2[i]->y <= 0 || coinsprite2[i]->y + coinsprite2[i]->height + wall\_2\_height >= winHeight)

coinsprite2[i]->dy = -coinsprite2[i]->dy;

rectmarie.x = marie\_x;

rectmarie.y = marie\_y;

rectmarie.w = marie\_width;

rectmarie.h = marie\_height;

rectcoin2[i].x = coinsprite2[i]->x;

rectcoin2[i].y = coinsprite2[i]->y;

rectcoin2[i].w = coinsprite2[i]->width;

rectcoin2[i].h = coinsprite2[i]->height;

if (collision(rectcoin2[i], rectmarie) == 1)

{

score -= 1;

coinsprite2[i] = NULL;

}

}

}

}

else if (id == 10)

{

if (numcoin2 < maxcoin2)

numcoin2++;

else

numcoin2 = 0;

if (coinsprite2[numcoin2] == NULL)

{

coinsprite2[numcoin2] = (sprite\*)malloc(sizeof(sprite));

coinsprite2[numcoin2]->width = 20;

coinsprite2[numcoin2]->height = 20;

coinsprite2[numcoin2]->x = rand() % (winWidth - coinsprite2[numcoin2]->width);

coinsprite2[numcoin2]->y = rand() % (winHeight - wall\_2\_height - coinsprite2[numcoin2]->height);

coinsprite2[numcoin2]->dx = 3;

coinsprite2[numcoin2]->dy = 1;

}

}

else if (id == 11 && score >= 0)

{

bad\_snake\_y += dy\_bad\_snake;

if (bad\_snake\_y <= 0 || bad\_snake\_y >= 100)

dy\_bad\_snake \*= -1;

rectmarie.x = marie\_x;

rectmarie.y = marie\_y;

rectmarie.w = marie\_width;

rectmarie.h = marie\_height;

rectbad\_snake.x = bad\_snake\_x;

rectbad\_snake.y = bad\_snake\_y;

rectbad\_snake.h = bad\_snake\_height;

rectbad\_snake.w = bad\_snake\_width;

if (collision(rectbad\_snake, rectmarie) == 1)

{

score -= 5;

heart -= 1;

marie\_x += 10;

}

}

if (id == 12 && score >= 100)

{

bad\_mush\_x += dx\_bad\_mush;

bad\_mush\_y += dy\_bad\_mush;

if (bad\_mush\_x <= 0 || bad\_mush\_x + bad\_mush\_width >= winWidth)

dx\_bad\_mush \*= -1;

if (bad\_mush\_y <= 0 || bad\_mush\_y + bad\_mush\_height + wall\_2\_height >= winHeight)

dy\_bad\_mush \*= -1;

rectmarie.x = marie\_x;

rectmarie.y = marie\_y;

rectmarie.w = marie\_width;

rectmarie.h = marie\_height;

rectbad\_mush.x = bad\_mush\_x;

rectbad\_mush.y = bad\_mush\_y;

rectbad\_mush.h = bad\_mush\_height;

rectbad\_mush.w = bad\_mush\_width;

if (collision(rectbad\_mush, rectmarie) == 1)

{

score -= 10;

heart -= 3;

marie\_x += 10;

}

}

paint();

return 0;

}

**4.8 NPC movement**

To control Mario via keyboard, we first use the function below in the Setup function to register keyboard event.

registerKeyboardEvent(keyEvent);

Then, we design a function using switch case to change the coordinate of Mario. Also, it should be able to change the coordinate of background to let the background move when Mario moves. Below are part of it.

**CODE:**

**case** VK\_UP:

Mario\_y -= 7;

**if** (Mario\_y <= 0)

Mario\_y = 0;

**break**;

**case** VK\_LEFT:

Mario\_x -= 7;

**if** (Mario\_x <= (600 + 40) && Mario\_x <= (600 + 40 - 6) && Mario\_y >= 175)

Mario\_x = 600 + 40;

**if** (Mario\_x <= 0)

Mario\_x = 0;

**if** (score <= 60 && Mario\_x > 0 && background\_1\_x < 0)

{

background\_1\_x += 5;

greenpole\_x += 5;

beginPaint();

putImageScale(&background\_1\_img, background\_1\_x, background\_1\_y, background\_1\_width, background\_1\_height);

putImageScale(&greenpole\_img, greenpole\_x, greenpole\_y, greenpole\_width, greenpole\_height);

endPaint();

}

**if** (score > 60 && Mario\_x > 0 && background\_2\_x < 0)

{

background\_2\_x += 5;

greenpole\_x += 5;

beginPaint();

putImageScale(&background\_2\_img, background\_2\_x, background\_2\_y, background\_2\_width, background\_2\_height);

putImageScale(&greenpole\_img, greenpole\_x, greenpole\_y, greenpole\_width, greenpole\_height);

endPaint();

}

**break**;

**4.9 Scene change**

At the very beginning, to let the game start after the start button is clicked, we design a mouseevent function. First, register it in the Setup function as follows.

**CODE:** registerMouseEvent(mouseEvent);

Then, the body of this function let timers start and initialize the original coordinate of Mario, backgrounds, and NPC after the click of start.

**CODE:**

**int** mouseEvent(**int** x, **int** y, **int** button,**int** event)

{

**if** (button == LEFT\_BUTTON && event == BUTTON\_DOWN && x > 275 && x < 425 && y > 100 && y < 160)

{

timerStart();

score = 0, heart = 10;

Mario\_x = 10, Mario\_y = 200;

bad\_missile\_x = 930, bad\_missile\_y = 80;

bad\_mask\_x = 0, bad\_mask\_y = 120;

background\_1\_x = 0, background\_1\_y = 0;

background\_2\_x = 0, background\_2\_y = 0;

bad\_green\_turtle\_x = 200, bad\_green\_turtle\_y = 190;

greenpole\_x = 600, greenpole\_y = 160;

bad\_snake\_x = 600, bad\_snake\_y = 0;

bad\_mush\_x = 0, bad\_mush\_y = 0;

}

*//else return;*

**return** 0;

}

After the player reaches the score of 60, the game enters another level and the background scene changes too.

**CODE:**

**if** (score > 60)

{

setTextBkColor(RGB(83, 53, 117));

setTextColor(WHITE);

putImageScale(&background\_2\_img, background\_2\_x, background\_2\_y, background\_2\_width, background\_2\_height);

}

If the player reaches the score of 150, the game ends as he wins. At that time, all the timers will be closed and the game will switch to the end screen.



**CODE:**

**if** (score >= 150)

{

fopen\_s(&fp1,"score.txt","at+");

fprintf(fp1,"%d",score);

fclose(fp1);

timerCancel();

putImageScale(&background\_2\_img, background\_2\_x, background\_2\_y, background\_2\_width, background\_2\_height);

putImageTransparent(&win\_img, 250, 50, 200, 60, BLACK);

putImageTransparent(&restart\_img, 275, 100, 150, 60, BLACK);

}

If the HP of Mario is not above zero, the game ends too as the player loses. At that time, all the timers will be closed and the game will switch to the end screen.



**CODE:**

**if** (heart <= 0)

{

fopen\_s(&fp1, "score.txt", "at+");

fprintf(fp1, "%d", score);

fclose(fp1);

timerCancel();

putImageScale(&background\_2\_img, background\_2\_x, background\_2\_y, background\_2\_width, background\_2\_height);

putImageTransparent(&lose\_img, 250, 50, 200, 60, BLACK);

putImageTransparent(&restart\_img, 275, 100, 150, 60, BLACK);

}

**5.Testing & Debuging**

**5.1 Heart and score will strangely change in several places.**

|  |  |  |
| --- | --- | --- |
| Situation | Expected Result | Real Result |
| Move Mario in all the places in the map | If Mario doesn’t collide with NPC or coins, the heart and socore are not supposed to change. | At some places, the heart and score will change even though not collide with NPC visually |

At first, when even didn’t know what went wrong. Every single thing seemed to be irrelevant. However, later, we found that although it’s not the time for some NPC like the snake and mask to appear, the collision judgement of those NPC still worked. When Mario happened to go by, the judgement of collision functioned, which resulted in the strange change of heart and score.

**5.2 The background of pictures adopted in the game is not transparent.**

|  |  |  |
| --- | --- | --- |
| Situation | Expected Result | Real Result |
| Playing the game | They shouldn’t have colorful backgrounds. | They have |

This problem plagued us for a long time. At first, we found that the format of jepg didn’t support transparent background. As a result, we changed it to the format of bmp and used GIMP to make it transparent. Nonetheless, this also failed. After pondering on it for a long time, we found that there was a function in our library that satisfied our demand. But, to our disappointment, the recognition of background color was not so good. When using GIMP to zoom in, we finally found that it’s colors that similar to the background color caused the problem above. In the normal size, it’s almost the same as the background color - that’s why the recognition of background color was not so good. At last, we utilized GIMP to solve this problem.

**5.3 The background can move out of the window.**

|  |  |  |
| --- | --- | --- |
| Situation | Expected Result | Real Result |
| Testing whether the background will move when Mario moves | If the boundary of background and that of the window overlap with each other, the background will not move | It moves anyway |

We pinned down the problem quickly - there must be something wrong with the logic of this judgement. After designing the logic again on paper, we translated it into code and the problem was conquered.

**5.4 The combination of interface and game scene.**

|  |  |  |
| --- | --- | --- |
| Situation | Expected Result | Real Result |
| Combining the interface with the game scene | If “start” is not clicked, the game shouldn’t start. | It starts even when the start button is not clikced. |

This problem caused a lot of trouble to us. We spent so much time trying to fix it, but all the efforts failed. Nonetheless, after judging the situation and problem calmly, we finally found where the problem lied in. There was something wrong with our logic. Our game used timer to help paint all the characters and coins. At first, our code let the timer start whenever it ran. Consequently, even when we didn’t click the start button, the background, the coins, and all the NPC would be painted. After changing the position of relevant code, we solved the problem successfully.

**6.Result & Conclusion**

In this semester, when doing this project, we found it quite painful, because we even didn’t know how to make the interface and how to control the role via keyboard. Consequently, all of us devoted a lot and spent much time in it. We worked together to decide the game to make, design the whole game, and pin down those strange bugs in our game. After finishing it, when we look back, we find it quite meaningful. Our game has a beautiful interface and we can also use keyboard to control the movement of Mario. Moreover, we have different NPC and three kinds of coins moving in different patterns. The process to achieve that enabled us to learn a lot. The process itself is just like climbing a high mountain. When plodding up along the ridge, with the sun hanging in the vault of azure sky above, one may sweat profusely. However, after reaching the summit, he could have every marvelous thing in his eyes. He could behold the white sea of clouds, the magnificent forest of mountains, and so forth. As for the things we have learnt, firstly, it’s crucial to make best of online resources. There are so many things we don’t understand and, on the Internet, there are always numerous videos and professional answerers waiting for us. With the assistance of those resources, most of our problems were successfully solved. Secondly, in a project, collaboration outweighs many other things. Someone has to allocate the responsibility properly. In our group, after choosing what we would make, Di Hou allocated all the things we need to do at the very beginning. We discussed the design of the whole game together. Wangze Shen was mainly for the logic and algorithm and he also did some coding. Di Hou was mainly responsible for coding and debugging and he also made contribution to the design of logic. Yuqi Liu and Mengya Han were for the processing of pictures and the design of interface. They also helped find some errors in our logic and did some debugging. Thirdly, the ability to judge and select suggestions is also of vital importance. When confronted with plights, different people could offer different remedies. Some may be right while some might be wrong. Having the sharp eyes to judge those solutions is the only road to find the way out.

In a nutshell, we managed to complete our game and learnt many precious things on our road to the destination.