

# 1. 如何在不影响定时执行的情况下, 进行输入操作

---

## 使用场景:

如贪吃蛇, 俄罗斯方块这类 自动操作+用户输入的游戏

## 涉及函数:

`int _kbhit();`

函数作用: 判断用户是否有输入(缓冲区是否有值)

返回值:

0: 表示用户没有输入(缓冲区没有值)

非0: 表示用户有输入内容(缓冲区有值)

头文件: `#include <conio.h>`

## 使用案例

```
int count = 1;
char input;
while (1) {
    count++; //计数器
    sleep(100);

    //每五次循环执行一次这里的操作
    if (count == 5) {
        // 每500毫秒打印一次---

        printf("---\n");
        count = 1; //初始化计数器
    }
    //当用户有输入时 执行这里的操作
    if (_kbhit()) {

        //如果输入的是esc 结束循环, 否则打印输入的内容
        input = _getch();
        if (input == 27) break;
        printf("%c", input);
    }
}
```

---

## 2. 如何在控制台中移动光标

---

## 使用场景:

贪吃蛇的蛇打印, 俄罗斯方块的下落方块打印. 界面边框打印, 局部清空等等

## 涉及函数:

### BOOL SetConsoleCursorPosition(句柄, 坐标);

句柄通过getStdHandle来获取即可, 所以我们只要知道坐标怎么传入就好  
坐标是结构体类型COORD 包含x和y属性 分别对应控制台的x坐标和y坐标  
该函数可以将光标移动到指定坐标位置

头文件: #include<windows.h> // (是的没错 mac系统没法使用)

## 使用案例

```
// 传入坐标, 将光标移动到指定坐标
void gotoXY(int x, int y)
{
    COORD c;
    c.X = x - 1;
    c.Y = y - 1;
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), c);
}

//使用案例
//清理指定矩形区域(从(x,y)到(x+w,y+h)的矩形
void clear(int x, int y, int w, int h)
{
    for (int i = 0; i < h; i++) {
        gotoXY(x, y + i);
        for (int j = 0; j < w; j++) putchar(' ');
    }
}

//打印边框
void printBox(int x, int y, int w, int h)
{
    //左竖线
    for (int i = 0; i < h; i++) {
        gotoXY(x, y + i);
        putchar('#');
    }
    //右竖线
    for (int i = 0; i < h; i++) {
        gotoXY(x + w, y + i);
        putchar('#');
    }
    //上横线
    gotoXY(x, y);
    for (int i = 0; i < w; i++) {
        putchar('#');
    }
    //下横线
```

```

gotoXY(x, y+h-1);
for (int i = 0; i < w; i++) {
    putchar('#');
}

//光标挪到其它位置 避免后面的输出覆盖这里的打印
gotoXY(x + w, y + h+1);

}

```

### 3. 如何设置控制台的字体颜色

#### 使用场景:

大富翁的不同玩家颜色标识, 贪吃蛇的食物闪烁, 菜单选中效果等等

#### 涉及函数:

##### BOOL SetConsoleTextAttribute(句柄, 颜色);

同上, 句柄是什么意思不用管 通过GetStdHandle获取即可.

颜色是用十六进制数表示, 如: 0x12

不同数字表示不同颜色

- 0: 黑
- 1: 蓝
- 2: 绿
- 3: 蓝绿
- 4: 红
- 5: 紫
- 6: 黄
- 7: 白

另外 +8表示高亮, 然后低位表示字体颜色, 高位表示背景颜色

例如: 0x2c的含义是

低位c表示4+8(十六进制) 也就是字体红色并且高亮

高位2表示背景绿色

头文件: #include<windows.h> // (是的没错 mac系统没法使用)

#### 使用案例

```

void setPrintColor(int color) {
    SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), color);
}

int main(){
    int userChoose = 0;
    while (1) {
        // ----- 打印界面 -----
        gotoXY(60, 3);
        printf("大    富    翁");
    }
}

```

```

        if (userChoose == 0) setPrintColor(0x6f);
        gotoXY(64, 6);
        printf("开始游戏");
        if (userChoose == 0) setPrintColor(0x0f);

        if (userChoose == 1) setPrintColor(0x6f);
        gotoXY(65, 9);
        printf("排行榜");
        if (userChoose == 1) setPrintColor(0x0f);

        if (userChoose == 2) setPrintColor(0x6f);
        gotoXY(64, 12);
        printf("游戏设置");
        if (userChoose == 2) setPrintColor(0x0f);

        if (userChoose == 3) setPrintColor(0x6f);
        gotoXY(64, 15);
        printf("退出游戏");
        if (userChoose == 3) setPrintColor(0x0f);

// ----- 接收用户输入 -----
char input = _getch();

// -----判断是方向上下还是回车-----
switch (input) {
case 'w':
    userChoose -= 1;
    if (userChoose == -1) userChoose = 3;
    break;
case 's':
    userChoose = (userChoose + 1) % 4;
    break;
case '\r':
    clear(3, 2, 80, 20);
    switch (userChoose) {
    case 0:
        //choosePlayer(); 调用游戏界面函数
        break;
    case 1:
        // 调用排行榜界面函数
        break;
    case 2:
        // 调用设置界面函数
        break;
    case 3:
        exit(0);
        break;
    }
    clear(3, 2, 36, 20);
    break;
}
}
return 0;
}

```

## 4. 如何播放音乐

### 使用场景:

游戏的音效, 背景音乐. 音乐播放器.

### 涉及函数:

`mciSendString(MCI指令, 0, NULL, NULL);`

#### 常用指令:

"open 文件路径 alias 别名 type mpegvideo"

表示将指定文件 视作mpegvideo类型打开 并且给一个别名 后续可以通过操作别名来控制音乐 (设置音量, 循环播放等等)

"play 文件路径或别名"

播放音乐

"play 别名 repeat"

循环播放音乐

"setaudio 别名 volume to 八进制数字"

设置音量大小

#### 头文件:

```
#include<windows.h>
```

```
#include<Mmsystem.h>
```

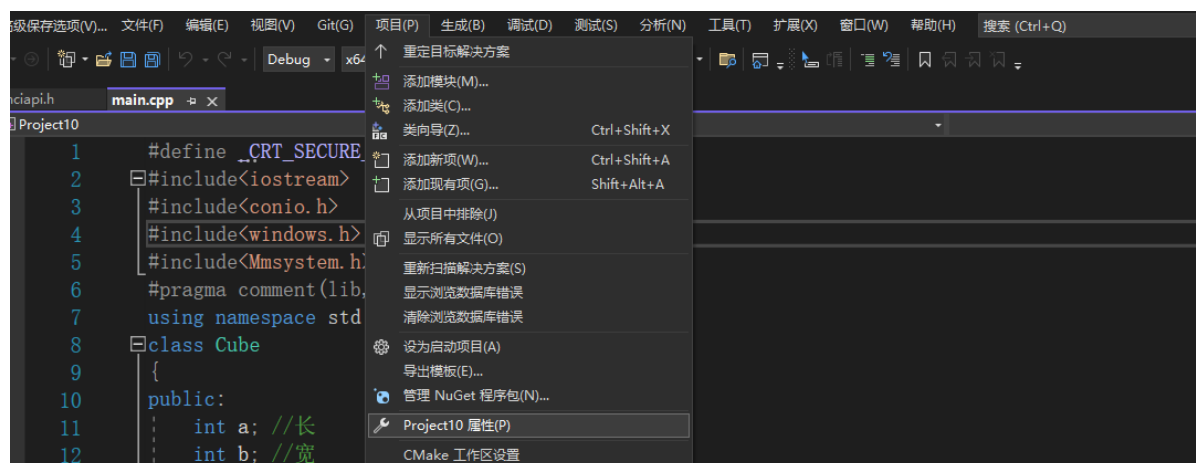
(注意 windows.h要在mmsystem.h的上面)

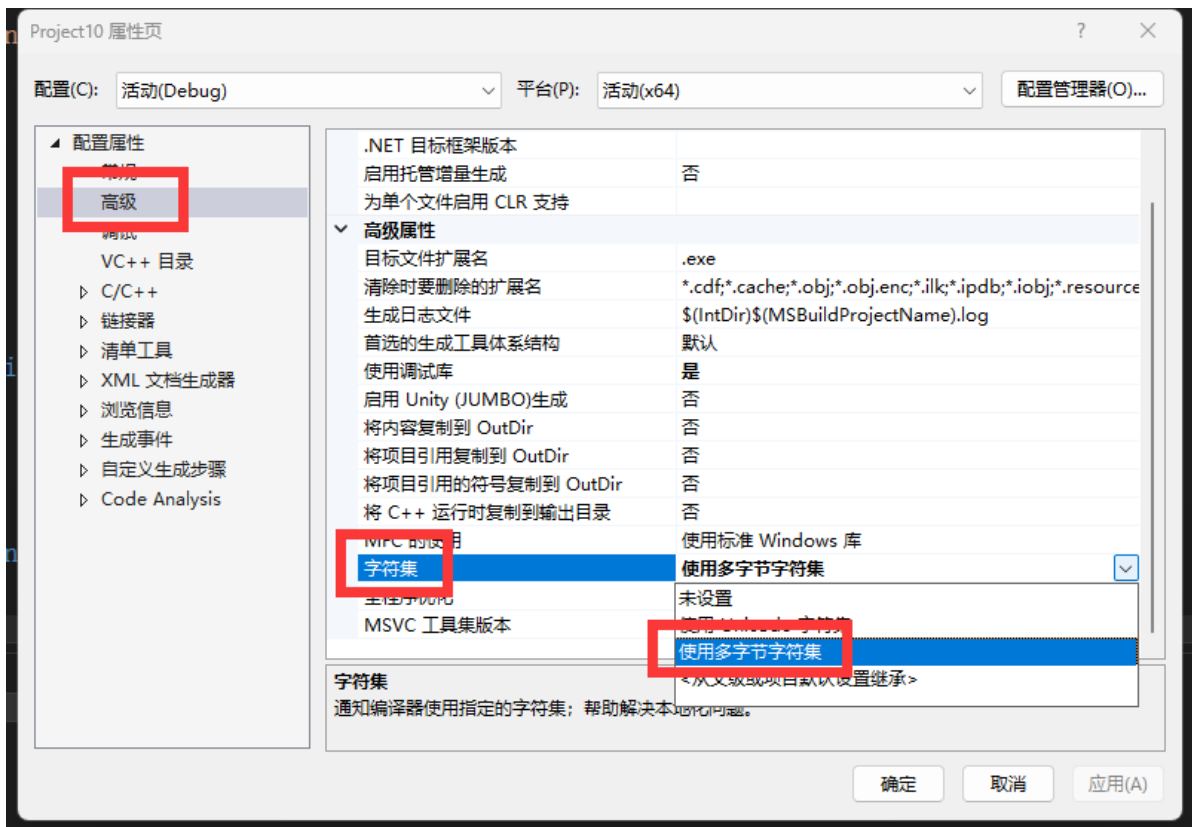
并且另外要导入winmm库:

```
#pragma comment(lib,"winmm.lib")
```

### 注意事项

1. 如果是直接传递字符串参数, 参数前面要加L 或者设置字符集为"使用多字节字符集"





2. 路径尽量避免中文, 特殊符号, 空格
3. 程序结束后会停止播放 可以加一个Sleep(10000)或者getchar()来卡住程序避免结束
4. 有的音乐格式不支持 可以用格式工厂转换为wav格式

## 使用案例:

```
#include<windows.h>
#include<Mmsystem.h>
#pragma comment(lib,"winmm.lib")

#define music_background 0
#define music_over 1
#define music_score 2
#define music_title 3
// 是否播放音乐 0表示不播放, 1表示播放
extern int isMusic=1;

//音乐文件的路径
const wchar_t musicPath[4][128] = {
    L"./background.wav",
    L"./music/over.wav",
    L"./music/score.wav",
    L"./music/title.wav"
};

// 播放指定音乐
int playMusic(int choose) {
    if (!isMusic) return 0;
    wchar_t cmd[MAX_PATH + 10];
    if(choose >=0 && choose<= 3){
```

```

        if(choose == music_title){
            // 如果是背景音乐 则循环播放
            wsprintf(cmd, L"open %s alias title_bgm type mpegvideo",
musicPath[choose]);
            mciSendString(cmd, 0, NULL, NULL);
            mciSendString(L"play title_bgm repeat", NULL, 0, NULL);
        }
        else if(choose == music_background){
            // 如果是背景音乐 则循环播放
            wsprintf(cmd, L"open %s alias background_bgm type mpegvideo",
musicPath[choose]);
            mciSendString(cmd, 0, NULL, NULL);
            mciSendString(L"play background_bgm repeat", NULL, 0, NULL);
        }
        else {
            // 否则 只播放一次
            wsprintf(cmd, L"play %s ", musicPath[choose]);
            mciSendString(cmd, 0, NULL, NULL);
        }
        return 1;
    }

    return 0;
};

// 停止播放指定音乐
int stopMusic(int choose) {
    wchar_t cmd[MAX_PATH + 10];
    if (choose == music_background || choose == music_title) {

    }
    if (choose == music_title) {
        return mciSendString(L"stop title_bgm", NULL, 0, NULL);
    }
    else if (choose == music_background) {
        return mciSendString(L"stop background_bgm", NULL, 0, NULL);
    }
    wsprintf(cmd, L"stop %s", musicPath[choose]);
    return mciSendString(cmd, NULL, 0, NULL);
}

```