

## Mutant Space Apes

If you like top down shooters that you aim with your mouse, then Mutant Space Apes is for you. Because that's what it is. Move with the arrow keys and try to avoid the ever spawning army of mutant apes while firing nuclear bananas at them. Each ape will take a few hits before they go down. But your superior aim and endless supply of bananas could allow you keep the upper hand.

For a time.

Mutant Space Apes is perhaps the simplest an arena mouse to aim shooter could be, which is very good if you ever wanted to make your own. A challenge for this game would be to rewrite it with multiple enemy types, shot powerups, and health pickups. But even as simple as it is Mutant Space Apes is balanced enough that it can keep you entertained.

Mutant Space Apes is by Jonatan Hedborg, created as an entry for MinorHack.

// mutantapes.cpp listing begins:

```
#include <cstdlib>
#include <ctime>
#include <cmath>
#include <list>
#include <allegro.h>

#define PLAYER_SPEED 2.0
#define ENEMY_SPEED 1.5
#define ROF 20
#define BANANA_SPEED 4
#define SPAWN_TIMER 150

using namespace std;

volatile int counter;
void my_timer_handler()
{
    counter++;
}
END_OF_FUNCTION(my_timer_handler)

struct OBJECT {
    float x,y,xv,yv;
    float angle;
    float health;
    float radius;

    OBJECT() {
        x = 0;
        y = 0;
        xv = 0;
        yv = 0;
        angle = 0;
        health = 0;
        radius = 0;
    }

    OBJECT(float ix, float iy, float ixv, float iyv, float iangle, float ihealth,
float iradius) {
        x = ix;
        y = iy;
        xv = ixv;
        yv = iyv;
        angle = iangle;
        health = ihealth;
        radius = iradius;
    }
};

bool isHit(OBJECT *o1, OBJECT *o2) {
    if( (o1->x - o2->x)*(o1->x - o2->x) + (o1->y - o2->y)*(o1->y - o2->y) < (o1->radius+o2->radius)*(o1->radius+o2->radius) ) return true;
    else return false;
}

int main() {
    srand(time(0));

    allegro_init();
    set_color_depth(32);
    set_gfx_mode(GFX_AUTODETECT_WINDOWED, 800, 600,0,0);
```

// Listing continued on next page...

// Listing continued from previous page

```
BITMAP *buffer = create_bitmap(800,600);
BITMAP *enemy_b, *player_b, *banana_b;

player_b = create_bitmap(40,40);
clear_to_color(player_b,makecol(255,0,255));
circlefill(player_b,20,20,15,makecol(163,110,70));

enemy_b = create_bitmap(30,30);
clear_to_color(enemy_b,makecol(255,0,255));
circlefill(enemy_b,15,15,12,makecol(37,154,14));
circlefill(enemy_b,20,8,3,makecol(255,255,255));
circlefill(enemy_b,20,22,3,makecol(255,255,255));
circlefill(enemy_b,20,8,1,makecol(0,0,0));
circlefill(enemy_b,20,22,1,makecol(0,0,0));

banana_b = create_bitmap(15,5);
clear_to_color(banana_b,makecol(223,230,25));
rectfill(banana_b,0,0, 3,4, makecol(95,56,14));
rectfill(banana_b,10,0, 14,4, makecol(95,56,14));
rectfill(banana_b,3,3, 10,5, makecol(255,0,255));
rectfill(banana_b,6,2, 7,5, makecol(255,0,255));

install_keyboard();
install_mouse();

LOCK_VARIABLE(counter);
LOCK_FUNCTION(my_timer_handler);

install_int_ex(my_timer_handler,BPS_TO_TIMER(60));

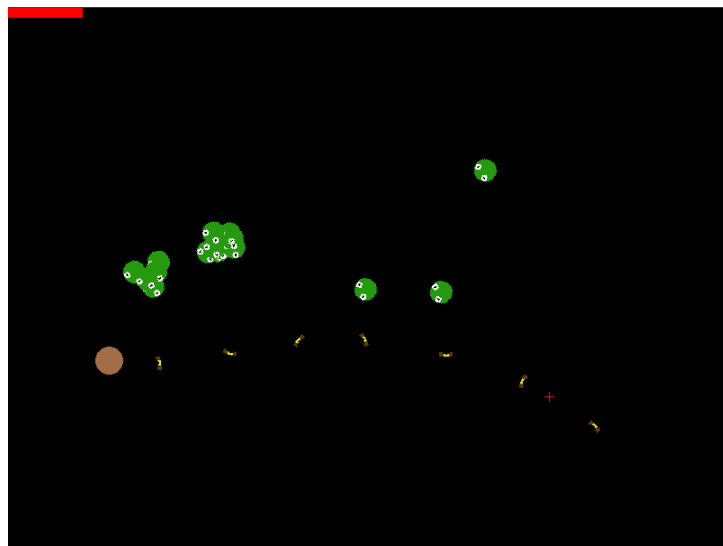
list<OBJECT> enemies;
list<OBJECT> bananas;
OBJECT player(400,300,0,0,0,200,15);
int fire_delay = 0;
bool gameOver = false;
int killCount = 0;
int timeToSpawn = 0;

while(!key[KEY_ESC] && !gameOver) {
    while(counter > 0) {
        timeToSpawn--;

        if( rand()%1000 < killCount || timeToSpawn < 0 ) {
            enemies.push_back(OBJECT(rand()%800,rand()%600,0,0,0,3,15));
            timeToSpawn = SPAWN_TIMER;
        }

        counter--;
        fire_delay--;

        if( key[KEY_UP] ) {
            player.y -= PLAYER_SPEED;
        }
        if( key[KEY_DOWN] ) {
            player.y += PLAYER_SPEED;
        }
        if( key[KEY_LEFT] ) {
            player.x -= PLAYER_SPEED;
        }
        if( key[KEY_RIGHT] ) {
            player.x += PLAYER_SPEED;
        }
    }
}
```



// Listing continued on next page...

// Listing continued from previous page

```
    }
    if( mouse_b && fire_delay < 1 ) {
        fire_delay = ROF;
        float angle = atan2(mouse_y-player.y, mouse_x-player.x);
        bananas.push_back(OBJECT(player.x,player.y,cos(angle)
*BANANA_SPEED,sin(angle)*BANANA_SPEED,0,1,5));
    }
    if( player.health < 0 ) {
        gameOver = true;
    }

    for(list<OBJECT>::iterator it = enemies.begin(); it != enemies.end
()); {
        if( (*it).health > 0 ) {
            float angle_to_player = atan2(player.y-(*it).y,player.x-
(*it).x);

            (*it).x += cos(angle_to_player)*ENEMY_SPEED;
            (*it).y += sin(angle_to_player)*ENEMY_SPEED;
            (*it).angle = angle_to_player;

            if( isHit(&(*it),&player) ) {
                player.health-=1;
            }

            it++;
        } else {
            killCount++;
            it = enemies.erase(it);
        }
    }

    for(list<OBJECT>::iterator it = bananas.begin(); it != bananas.end
()); {
        if( (*it).x > -50 && (*it).x < 850 && (*it).y > -50 && (*it).y <
650 && (*it).health > 0 ) {
            (*it).x += (*it).xv;
            (*it).y += (*it).yv;
            (*it).angle += 0.1;

            for(list<OBJECT>::iterator eit = enemies.begin(); eit != ene-
mies.end(); eit++) {
                if(isHit(&(*it),&(*eit))) {
                    (*eit).health -= 1;
                    (*it).health = 0;
                }
            }

            it++;
        } else {
            it = bananas.erase(it);
        }
    }

}

clear(buffer);

for(list<OBJECT>::iterator it = bananas.begin(); it != bananas.end();
it++) {
    rotate_sprite(buffer,banana_b,(*it).x-7,(*it).y-3,ftofix
```

// Listing continued on next page...

// Listing continued from previous page

```
((*it).angle*128/M_PI));
    }

    for(list<OBJECT>::iterator eit = enemies.begin(); eit != enemies.end();
eit++) {
        rotate_sprite(buffer,enemy_b,(*eit).x-7,(*eit).y-3,ftofix
(((*eit).angle*128/M_PI));
    }

    rotate_sprite(buffer,player_b,player.x-20,player.y-20,ftofix
(player.angle*128/M_PI));

    line(buffer,mouse_x-5,mouse_y,mouse_x+5,mouse_y,makecol(220,30,30));
    line(buffer,mouse_x,mouse_y-5,mouse_x,mouse_y+5,makecol(220,30,30));

    rectfill(buffer,0,0,player.health,10,makecol(255,0,0));

    vsync();
    blit(buffer,screen, 0,0,0,0, 800,600);
}

clear(screen);
textprintf_ex(screen,font,20,200,makecol(0,200,20),-1,"GAME OVER MAN, GAME
OVER!");
textprintf_ex(screen,font,20,212,makecol(0,200,20),-1,"You managed to killify
%i mutant apes",killCount);
textprintf_ex(screen,font,20,224,makecol(0,200,20),-1,"Press space to exit");

while(!key[KEY_SPACE]) {}

return 0;
}END_OF_MAIN()
```