Carlos Doblas Sánchez 1ºD
El desarrollo de la práctica se me ha hecho llevadero y he aprendido una serie de utilidades que consideraba secundarias a la hora de programar antes de hacer la práctica como son la modularización y la utilización de makefiles. Eso fue una de las primeras dificultades que tuve al principio de la práctica, aunque una vez entendido todo el proceso, me ha sido muy sencillo ir modularizando y me ha facilitado mucho las cosas. Otro apartado que me costó un poco de entender fue el tema del uso de la funciones de miniwin, pero mirando miniwin.h y probando un poco me manejé bastante fluido. Lo último a destacar que me supusiera alguna dificultad, fue el tema de memoria dinámica y la sobrecarga de operadores, pero por fallos tontos. Utilizando el valgrind no he tenido ningún problema.
Respecto a las tareas adicionales, en un primer momento tenía pensado hacer un menú para poder elegir distintas modalidades a hacer con las bolas, y así he hecho. El primer modo que pensé en hacer fue el de que las bolas mas rápidas fuesen eliminando a las bolas más lentas cuando chocasen, y fue lo primero que hice. No fue muy complicado ya que sólo tuve que implementar la función velocidad, que calculaba la velocidad que llevaba la partícula con las variables dx y dy, junto a la función colisión que devolvía si estaba colisionando una partícula con otra, y aparte la función desaparece, que pone la partícula a desaparecer a color negro, establece dx y dy en 0 para que no se mueva y por último establecia la partícula en las cordenadas (-1, -1) para que no interactuase con las otras partículas si implementaba otros método como el rebota.
Luego pensé en lo mismo pero utilizando algo relacionado con los colores. Con lo que implementé la misma dinámica con la diferencia que en vez de desaparecer, al colisionar, la más rápida le transferiría su color a la impactada más lenta. Para una mayor visibilidad del juego, también implementé el método rebota, para una vez que colisionaran, rebotasen entre ellas.
Por último y gracias a un amigo, me dijo que si podía manejar una pelota con el ratón, y mirando el miniwin.h, me di cuenta que tenía funciones que te devolvian la posición del ratón y otra que te decia si el ratón estaba dentro de la ventana de las particulas, y así lo hice. Probé un poco las funciones para mover la partícula en donde me ocurrió algo bastante curioso y es tuve un problema a la hora de configurarlo ya que la función raton_y () estaba mal implementada por defecto devolviendo también la posicion del x.

```
float raton_y() {
return _mouse_state.x;
}
```

Me metí en el cpp, me di cuenta de que estaba mal, cambié el .x por .y, y conseguí resolver el problema.
Y una vez conseguí mover la partícula con el ratón lo que me pareció mas interesante fue implementar una especie de juego en el que tuvieses que esquivar las bolas con el ratón.
Adicionalmente, he implementado un método de inicio, en el que las partículas se quedan en la pantalla hasta que se pulsa la tecla espacio para que desde que pulsas la opción del menú en el terminal hasta que abrías la pantalla donde están las partículas empezaban a interactuar sin poder verlas. Aparte de esto, he implementado bastantes mejoras cómo que cuando saques
el ratón de la ventana se quede en pausa el juego, ya que si sacabas el ratón, tu partícula se quedaba en un extremo de la pantalla y te chocaba otra partícula. También he implementado varios métodos de mejora de lectura de código.
En la implementación de mi programa, he implementado un archivo auxiliar llamado menu.h en el que incluyo varias funciones para la facilitación del menú aunque posteriormente tambien decidí implementar tres funciones auxiliares para la facilitación de lectura del código de interacción de partículas. Habría podido implementar muchas más cosas pero, esto creo que es suficiente para demostrar mis habilidades de programación y ver como me desenvuelvo con una biblioteca externa como es miniwin.h por la falta de tiempo debido a que también tengo que estudiar otras asignaturas.
Por lo tanto, trás analizar y valorar mi trabajo creo que me asignaría la nota de sobresaliente ya que creo que he implementado cosas curiosas y no he decidido implementar más cosas por falta de tiempo.

Por último, destacar que hay varios fallos de código en los programas de prueba, como funciones que no existen, variables inútiles o vectores que no están creados que se deben de corregir.
Lo siento por las memorias que te envié las hice al final y con mucha prisa y no me di cuenta de la gran cantidad de faltas de ortografía y de puntuación que había.

CODIGO FUENTE DE SIMULACIÓN

```
int main(){
srand(time(0));
int ancho = 1280, alto = 720, particulas = 20;
//VARIABLES PARA EL MENU
int opcion = 0;
bool salir = false;
//VARIABLES QUE QUE UTILIZO
Particula P(0,0);
Nubeparticulas nb(30);
mensaje_menu();
do{
cin >> opcion;
switch(opcion){
case 1:
vredimensiona(ancho, alto);
nb.pintar();
while(tecla() != ESPACIO){
espera(200);
}
while (salir == false && tecla()!= ESCAPE) {
while(raton_dentro()!= true){
espera(200);
}
borra();
P.Set_x(raton_x());
P.Set_y(raton_y());
nb.Mover(ancho,alto);
nb.pintar();
rebote(nb);
P.pinta_pelota();
for (int i = 0; i<nb.Get_utils(); i++){
if (P.Colision(nb.ObtieneParticula(i)) ){
P.desaparece();
salir = true;
}}
refresca();
espera(25);
}
vcierra();
break;
case 2:
vredimensiona(ancho, alto);
nb.pintar();
while(tecla() != ESPACIO){
espera(200);
}
while (salir == false && tecla()!= ESCAPE) {
borra();
nb.pintar();
change_color(nb);
rebote(nb);
nb.Mover(ancho,alto);
refresca();
espera(20);
}
vcierra();
```

```
break;
case 3:
vredimensiona(ancho, alto);
nb.pintar();
while(tecla() != ESPACIO){
espera(200);
}
while (salir == false && tecla()!= ESCAPE) {
borra();
nb.pintar();
elimina(nb);
nb.Mover(ancho,alto);
refresca();
espera(20);
}
vcierra();
break;
default:
cout<< "Cerrando el programa..." <<endl;
salir == true;vcierra();
break;
}
}while(salir == false);
return 0;
}
```