```cpp
#include <iostream>
#include <graphics.h>
#include <time.h>

#define CANVAS_WIDTH 1280
#define CANVAS_HEIGHT 720

typedef struct M2d{
    int x;
    int y;
};
typedef struct Ball{
    int r=10;
    int springConstant = 1;
    M2d position = { 0,0 };
    M2d velocity= { 0,0 };
    M2d acceleration= { 0,0 };
};
typedef struct EnityChain{
    Ball* ball;
    EnityChain* next;
} ;
typedef struct World{
    int gravity;
    M2d size;
    EnityChain* enityChain;
};

void* createNewBall(EnityChain **node, Ball ball) {
    if ((*node) == NULL) {
        EnityChain* newNode = (EnityChain*)malloc(sizeof(EnityChain));
        Ball* newBall = (Ball*)malloc(sizeof(Ball));
        *node = newNode;
        *newBall = ball;
        (*newNode).ball = newBall;
        (*newNode).next = NULL;
        return newNode;

    }
    else {
        return createNewBall(&((**node).next), ball);
    }
}
void boxBounce(Ball* ball, M2d boxSize) {
    int* x = &(*ball).position.x, * y = &(*ball).position.y,
        * accx = &(*ball).acceleration.x, * accy = &(*ball).acceleration.y;
    if (*x < 0)*accx += -*x / (*ball).springConstant;
    else if (*x > boxSize.x)*accx += (boxSize.x - *x) /
      (*ball).springConstant;
    if (*y < 0)*accy += -*y / (*ball).springConstant;
    else if (*y > boxSize.y)*accy += (boxSize.y - *y)/(*ball).springConstant;
}
void refreshVelocity(Ball* ball) {
    M2d *acc = &(*ball).acceleration,
        *vel = &(*ball).velocity;
    (*vel).x += (*acc).x;
```

```
56      (*vel).y += (*acc).y;
57      (*acc).x = 0;
58      (*acc).y = 0;
59      return ;
60  }
61  void refreshPosition(Ball* ball) {
62      M2d* vel = &(*ball).velocity,
63         * pos = &(*ball).position;
64      (*pos).x += (*vel).x;
65      (*pos).y += (*vel).y;
66  }
67  void PhysX(World *world) {
68      EnityChain *node = (*world).enityChain;
69      for (int i = 0; node != NULL; i++) {
70          Ball* ball = (*node).ball;
71          //Phyx start
72          boxBounce(ball,(*world).size);
73          refreshVelocity(ball);
74          refreshPosition(ball);
75          (*ball).acceleration.y += (*world).gravity;
76          //Phyx end
77          node = (*node).next;
78      }
79      return;
80  }
81  void Draw(World *world) {
82      cleardevice();
83      EnityChain* node = (*world).enityChain;
84      for (int i = 0; node != NULL; i++)
85      {
86          Ball* ball = (*node).ball;
87          fillcircle((*ball).position.x, (*ball).position.y, (*ball).r);
88          node = (*node).next;
89      }
90      return;
91  }
92
93  int main()
94  {
95      initgraph(CANVAS_WIDTH, CANVAS_HEIGHT);
96      std::cout << "Hello World!\n";
97      int ballCount = 5;
98      World mainWorld{
99          2,
100         {1000,400},
101         NULL
102     };
103     EnityChain** node = &(mainWorld.enityChain);
104     for (int i = 0; i < ballCount; i++) {
105         Ball ball = {
106             10,
107             1,
108             { 200 + rand() % (i + 10),200 + rand() % (i + 10) },
109             { rand() % (i + 10), rand() % (i + 10)},
110             { 0, 0 },
111         };
```

```
112             createNewBall(node, ball);
113         }
114         for (long frameCount=0;true;frameCount++)
115         {
116             Draw(&mainWorld);
117             PhysX(&mainWorld);
118             Sleep(16);
119         }
120         return 0;
121     }
```