Exercițiul 11

Clasa `MovablePoint` reprezintă un punct într-un sistem de coordonate `xOy` care se poate deplasa cu o viteză (`xSpeed, ySpeed`) (`xSpeed` unități pe axa `Ox` şi `ySpeed` unități pe axa `Oy`).

Să considerăm diagrama UML de clase din figura 1.

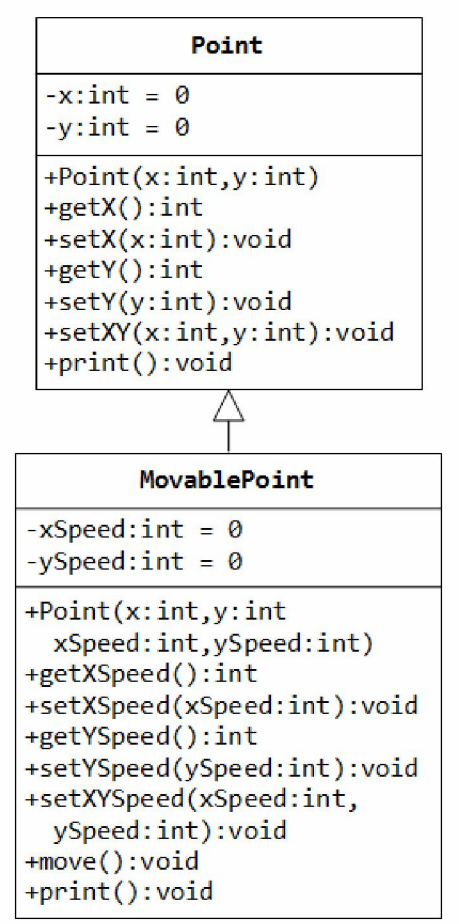

Figura 1

Clasa `MovablePoint` va conține următoarele elemente (a se vedea diagrama UML a clasei):

- două atribute private, `xSpeed` şi `ySpeed` de tipul întreg, şi care au valoarea implicită `0`, fiecare.
- constructor cu parametri, constructor de copiere, metode *getter* şi *setter* pentru atributele private.
- o metodă `setXYSpeed()` pentru a inițializa ambele valori `xSpeed` şi `ySpeed` ale unui punct care se deplasează.
- o metodă `move()` ce realizează deplasarea punctului corespunzător trecerii unei unități de timp (calculează noile coordonate `x` și `y` ale punctului).

- o metodă `print()` ce afişează "`(vx,vy), viteza=(vxspeed, vyspeed)`", unde `vx`, `vy`, `vxspeed` şi `vyspeed` sunt valorile atributelor `x`, `y`, `xSpeed`, `ySpeed`.
- După modelul de la clasa `Circle`, se doreşte să se specifice declaraţia clasei în fişierul `MovablePoint.h` iar definiţia metodelor clasei să fie realizată în fişierul `MovablePoint.cpp`.

Referinţe

Clasa `Point`.

Proiectul va contine 5 fişiere: `Point.h`, `Point.cpp`, `MovablePoint.h`, `MovablePoint.cpp` şi `test-points.cpp`.

Codul de testare pentru clasa `MovablePoint`:

```
int main() {
  Point p1(3, 3);

  p1.print();
  cout << endl;

  MovablePoint m1(3, 7);

  m1.print();
  cout << endl;

  m1.setXSpeed(8);
  m1.move();
  m1.print();
  cout << endl;

  MovablePoint m2(3, 7, 9, 11);

  m2.print();
  cout << endl;

  int time = 0;
  while (time < 5) {
    time++;
    m2.move();

    cout << "Dupa " << time << "s: ";
    m2.print();
    cout << endl;
  }

  cout << "Schimbam viteza si directia de deplasare\n";

  m2.setXYSpeed(-1, -3);
  while (time < 10) {
    time++;
    m2.move();

    cout << "Dupa " << time << "s: ";
    m2.print();
    cout << endl;
  }

  return  0;
}
```