PROGRAMACIÓN ORIENTADA A OBJETOS

EXAMEN 2

**1**. Dado el siguiente programa:

1. #include <iostream>
2. using namespace std;
3. class B{
4. public:
5. void f(){cout << "f() de B"<< endl;}
6. virtual void g() { cout << "g() de B" << endl;}
7. virtual void h()=0;
8. virtual ~B(){};
9. protected:
10. int b;
11. };
12. class D1: virtual public B{
13. public:
14. void f(){cout << "f() de D1"<< endl;}
15. virtual void g(){cout << "g() de D1" << endl;}
16. void h(){cout << "h() de D1"<<endl;}// Se debe de redifinir el virtual puro
17. protected:
18. int d1;
19. };
20. class D2: virtual public B{
21. public:
22. void f(int i){cout << "f(" << i <<") de D2" << endl;}
23. virtual void h(){ cout << "h() de D2 "<< endl;}
24. protected:
25. int d2;
26. };
27. class D3: public D1{
28. public:
29. void g(){cout << "g() de D3" << endl;}
30. void h(){ cout << "h() de D3" << endl;}
31. private:
32. int d3;
33. };
34. class D4: public D1,public D2{
35. public:
36. void h(){cout << "h() de D4" << endl;}
37. private:
38. int d4;
39. };
40. void f(B& b){
41. cout <<"f() externa" << endl;
42. b.f();
43. b.g();
44. b.h();
45. }
46. int main(){
47. B b;
48. B \*pB;
49. D1 d1;
50. D2 d2;
51. D3 d3;
52. D4 d4;
53. f(d1);
54. f(d2);
55. f(d3);
56. f(d4);
57. d4.D1::f();
58. d4.D2::f(5);
59. d4.D2::f(3.7);
60. d4.g();
61. d4.h();
62. pB=new D4;
63. pB->f();
64. pB->D4::f(3);
65. pB->g();
66. pB->h();
67. delete pB;
68. }
69. Corrija los errores, si los hay, en la definición de las clases B, D1, D2, D3 y D4. Para cada clase enumere

sus miembros declarados o definidos explícitamente

1. Diga si el programa provoca algún error de compilación o de ejecución y porqué. Si lo tiene, modifique el código adecuadamente para solucionarlo.

**2.** Dada la clase paramétrica:

template<class T1, class T2> class par{

public:

par(): prime(T1()), segun(T2()){}

par(const T1& x, const T2& y): prime(x), segun(y){}

T1 primero() const {return prime;}

T1& primero(){return prime;}

T2 segundo() const {return segun;}

T2& segundo(){return segun;}

private:

T1 prime;

T2 segun;

};

1. Sobrecargar operador << para la plantilla de clase par
2. Defina un tipo racional a partir de ella para representar número racionales. Sobrecargue el operador suma de nº racionales
3. Defina una clase complejo con parte real e imaginaria de tipo double. Sobrecargue el operador de autosuma (+=) de nº complejos