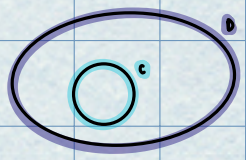
Ereditarietà:

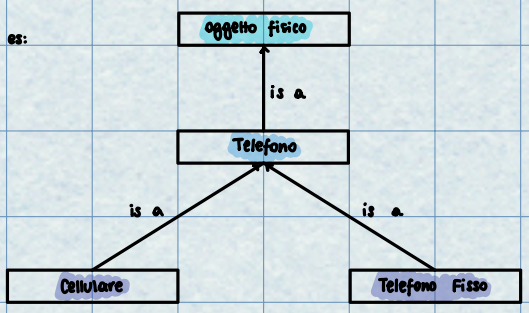
Classe: una classe C rappresenta l'insieme di tutti gli oggetti caratterizzati da un certo stato (proprietà)

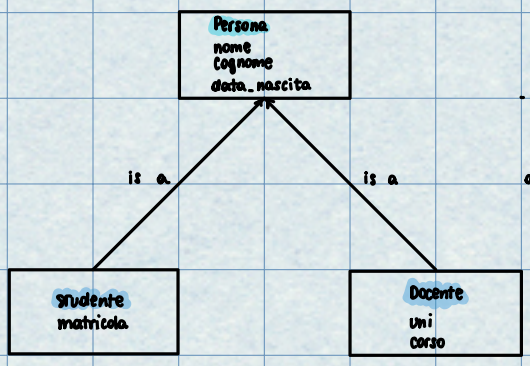
una classe C può essere sottoinsieme di una classe più ampia D e un oggetto della classe C apparterrà anche alla classe D



Classi, sottoclassi e superclassi

es:





. Studente e Docente sono sottoclassi di Persona

=> hanno tutte le caratteristiche di Persona

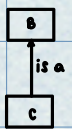
=> e anche tutti i metodi

Ereditarietà:

Una classe eredita tutte le caratteristiche della sua superclasse

Una sottoclasse si dice derivata dalla sua superclasse

in C++




```
class B {
   ...
};
```

```
class C : public B {
   ...
}
```

"la classe C è derivata dalla classe B"

la classe C è derivata in maniera pubblica dalla classe B

:gli attributi ereditati mantengono le stesse proprietà di visibilità della superclasse

Es C++: Punto

```
# include <iostream>

Using namespace std;
```

```
class Punto {
    protected:                    <-- visibili all'interno della classe e in tutte le sottoclassi, ma non all'esterno
        int x;
        int y;
    public:
        void setX (int x) {
            this -> x = x; }
        int getX () {
            return x; }
        void setY (int y) {
            this -> y = y; }
        int getY () {
            return y; }
        void print () {
            cout << "(" << x << ", " << y << ")"; }

class PuntoColorato : public Punto {
    private:
        String color;
    public:
        void setColor (String color) {
            this -> color = color; }
        String getColor () {
            return color; }
```

```
        void print () {
                cout <<"(" << x << ", " << y << ", " << colore << ")"; }


int main() {
        Punto p;
        p.setX(4);
        p.setY(2);
        p.print();
        PuntoColorato pc;
        pc.setColore("rosso");
        pc.setX(3);
        pc.setY(4);
        pc.print();
        return 0; }
```

```
class Persona {
        protected:
                string name;
                string Surname;
                string address;
        public:
                Persona() {
                        this->name = "xxx";
                        this->surname= "xxx";
```

```
class Studente : public Persona {
        private:
                int matricola;
                String email;
        public:
                Studente( string nome, String cognome, int matricola) : Persona (nome, cognome) {
                        this->matricola = matricola;
                        this->email = nome + "." + cognome + "@.unipr.it";}
```

```
        this.address = "xxx";}

    Persona (String name, String surname){
        this->name = name;
        this->surname = surname;
        this->address = "xxx";

    String  getName ()
        return name;}

    String  getSurname () {
        return surname;}

    String  getAddress(){
        return address;}

    void    setName (String name){
        this -> name = name;}

    void    setSurname (String surname){
        this ->surname= surname;}

    void    setAddress (String address){
        this-> address = address ;}

    void    stampa (ostream & dest) {
        dest << name << "  " << surname << endl;
        dest << indirizzo << endl; }

int main () {
Persona  p ("Vincenzo", "Arteri");
p.stampa (cout);
```

```
Studente () : Persona ("Marco", "Rossi")
        this->matricola = 0;          -> forzo ad usare il costr con parametri
        this->email = "xxx"; }

void stampa (ostream &  dest) {
        Persona :: Stampa (dest);
        cout << "matricola:  " << matricola << endl;
        cout << "mail: " << mail;}
```

```
Studente s ("Marco", "Rossi", 1234);

s.stampa (cout);

}
```