INSTITUTO FEDERAL
Fluminense
Campus Quissamã

MINISTÉRIO DA
EDUCAÇÃO

PÁTRIA AMADA
BRASIL
GOVERNO FEDERAL

Curso: Integrado Informática
Professor: Luiz César
Turma: 2° ano informática
Aluno: Elian Gonçalves e Cássia Vitória Gomes
Assunto: Herança (Java)
Disciplina: Técnicas de Programação

Herança na Programação Orientada a Objetos(Java)

A herança é uma forma de reutilização de software. Muitas das vezes as classes diferentes possuem características comuns, então utilizamos características de uma classe que já existe, ela absorve atributos e comportamentos que já existem e incluem os seus próprios, basicamente significa que uma classe herda atributos e métodos de uma classe “mãe”.
Para executar o conceito de herança em Java usa-se a palavra reservada “extends”
Exemplo:

```
public class Funcionario extends Pessoa {
private String matricula;
}
```

Java permite que uma classe herde apenas características de apenas uma classe, ou seja, não pode ter heranças múltiplas mas é permitido heranças em cadeias, por exemplo: **se a classe Mamífero herda a classe Animal, quando fizermos a classe Cachorro herdar a classe Mamífero, a classe Cachorro também herdará as características da classe Animal.**

Exemplo de código com Herança:

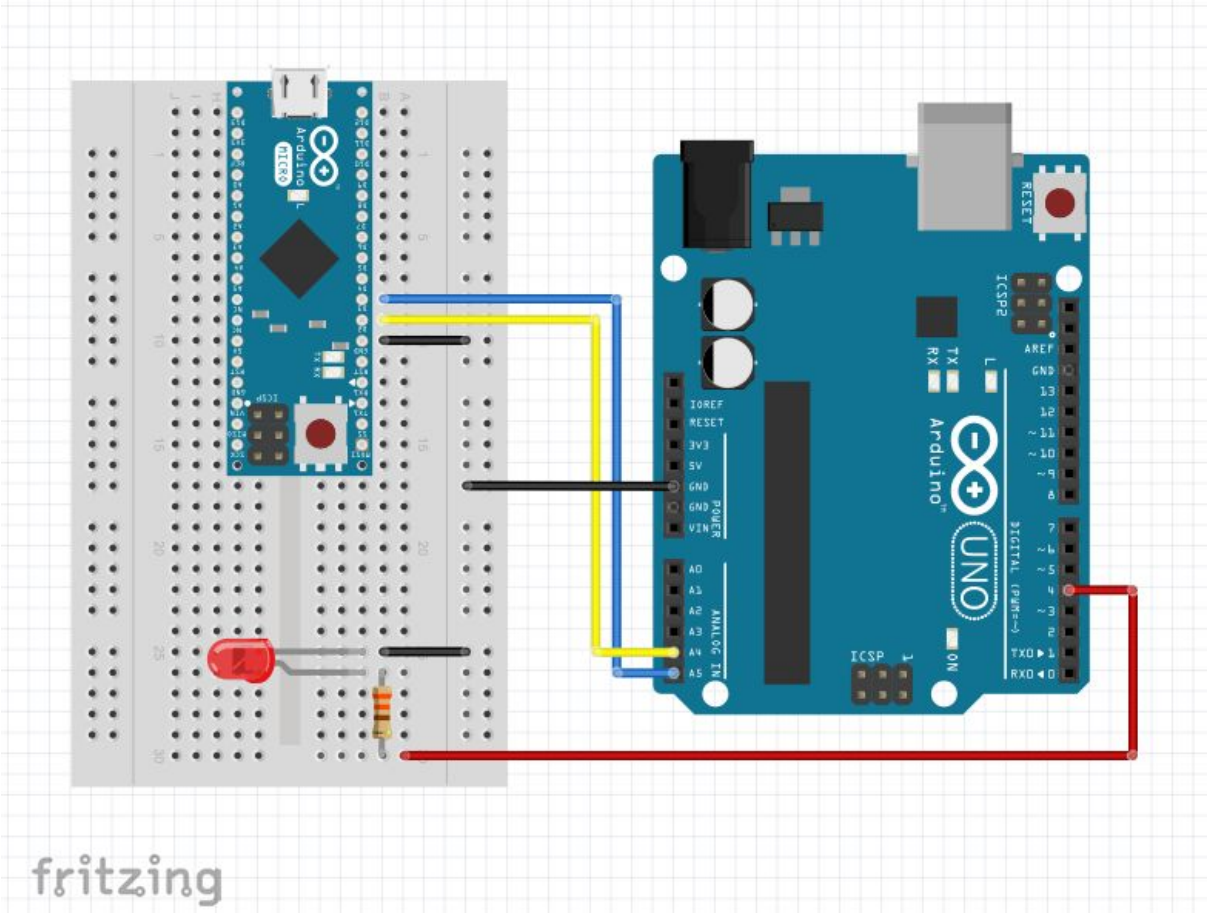
package heranca;
Pessoa.java

```
public class Pessoa {
  String nome,
  idade,
  endereco= "R: Java ,501"
  public void ImprimeNome(){
    System.out.println("o nome é:");
    System.out.println("Endereco: " + endereco);
    }
  }
```

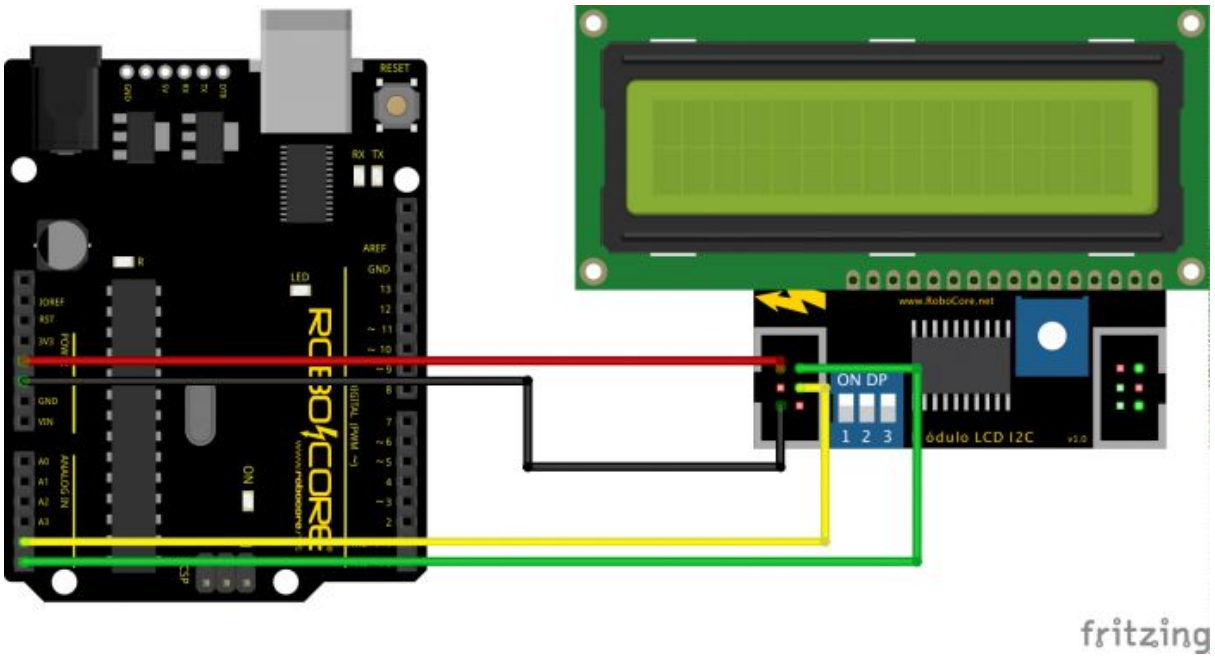
Fornecedor.java

```
package heranca;
Forncedor.java
public class Fornecedor extends Pessoa {
  String cnpj;
  public void ImprimeNome (){
    System.out.println("O nome do fornecedor é : " + nome + "\n Cnpj: " + cnpj);
  }
}
cliente.java
package heranca;
public class Cliente extends Pessoa {
  String cpf;
  public void ImprimeNome (){
    System.out.println("Nome do cliente é : " + nome + "\n Nº CPF: " + cpf + "\n Seu
endereco :" + endereco);
     }
}
Principal.java
package heranca;
public class Principal {
  public static void main(String[] args) {
    Cliente c = new Cliente()
    c.nome="Luiz";
    c.cpf="073.777.796-21";
    c.ImprimeNome();

    Fornecedor f = new Fornecedor ();

    f.nome="Deltatronic";
    f.cnpj="073.856.9856.52-10";

    f.ImprimeNome();
    }
}
```

Exemplos:



Na montagem deste hardware devemos garantir que os pinos SDA, SCL e GND de ambos os dispositivos estejam respectivamente conectados entre si. No Arduino UNO, os pinos SDA e SCL são os pinos analógicos A4 e A5 respectivamente, ao passo que, no Arduino Micro, estes são os pinos D2 e D3.



Código direcionado a imagem a cima.

```
 #include <Wire.h>
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x20,16,2);  // Criando um LCD de 16x2 no endereço 0x20

void setup()
{
  lcd.init();                  // Inicializando o LCD
  lcd.backlight();             // Ligando o BackLight do LCD
  lcd.print("Hello, world!"); // Exibindo no LED Hello, world!
}
void loop()
{
}
```

Referências:
nLink:https://www.robocore.net/tutorials/primeiros-passos-com-modulo-i2c.html
https://portal.vidadesilicio.com.br/i2c-comunicacao-entre-arduinos/
http://microcontrolandos.blogspot.com/2012/12/comunicacao-i2c.html