CONSTRUCTORS:

Why do we need constructors?

Initialization: Constructors initialize object properties when creating new objects.

Ensuring Valid State: They ensure objects start in a valid state with required values.

Connection & Inheritance: Constructors connect classes, aiding inheritance and object creation.

Example:

DEFAULT CONSTRUCTOR :

package JAVA\_CONCEPTS;

public class ApplicationForm {

private String name;

private String email;

private String phone;

public ApplicationForm() {

this.name = "HEMAMALANI";

this.email = "hema@2511.com";

this.phone = "3698574124";

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public String getPhone() {

return phone;

}

public void setPhone(String phone) {

this.phone = phone;

}

public static void main(String[] args) {

ApplicationForm applicationForm = new ApplicationForm();

System.out.println("Default Name: " + applicationForm.getName());

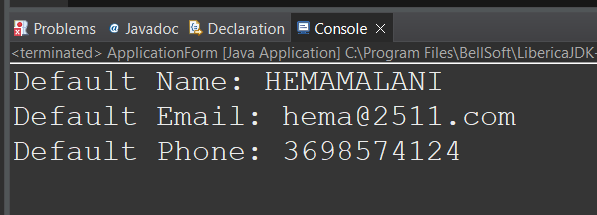
System.out.println("Default Email: " + applicationForm.getEmail());

System.out.println("Default Phone: " + applicationForm.getPhone());

}

}

OUTPUT:



PARAMETERIZED CONSTRUCTOR :

package JAVA\_CONCEPTS;

public class ApplicationForm {

private String name;

private String email;

private String phone;

public ApplicationForm(String name, String email, String phone) {

this.name = name;

this.email = email;

this.phone = phone;

}

public String getName() {

return name;

}

public String getEmail() {

return email;

}

public String getPhone() {

return phone;

}

public static void main(String[] args) {

ApplicationForm applicationForm = new ApplicationForm("John ", "john@123@gmail.com", "123-456-7890");

System.out.println("Name: " + applicationForm.getName());

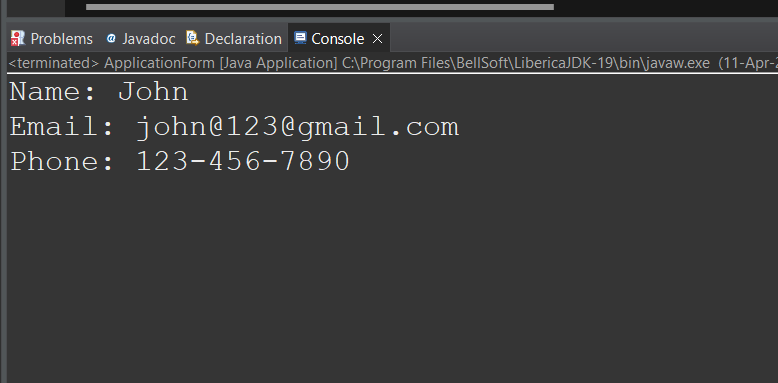
System.out.println("Email: " + applicationForm.getEmail());

System.out.println("Phone: " + applicationForm.getPhone());

}

}

OUTPUT:



COPY CONSTRUCTOR:

Making Copies: When we want to create a new object with the same values as an existing object, we use a copy constructor. It's like making a photocopy of something we already have.

Maintaining Original: Copy constructors ensure that changes to the new object don't affect the original object. Each object remains independent.

EXAMPLE:

package JAVA\_CONCEPTS;

public class ApplicationForm {

private String name;

private String email;

private String phone;

public ApplicationForm(String name, String email, String phone) {

this.name = name;

this.email = email;

this.phone = phone;

}

public ApplicationForm(ApplicationForm original) {

this.name = original.name;

this.email = original.email;

this.phone = original.phone;

}

public String getName() {

return name;

}

public String getEmail() {

return email;

}

public String getPhone() {

return phone;

}

public void setName(String name) {

this.name = name;

}

public void setEmail(String email) {

this.email = email;

}

public void setPhone(String phone) {

this.phone = phone;

}

public static void main(String[] args) {

ApplicationForm originalForm = new ApplicationForm("Hemamalani", "hema2511@gmail.com", "6325478954");

ApplicationForm copiedForm = new ApplicationForm(originalForm);

copiedForm.setName("Sathiya");

System.out.println("Original Form: " + originalForm.getName() + " " + originalForm.getEmail() + " " + originalForm.getPhone());

System.out.println("Copied Form: " + copiedForm.getName() + " " + copiedForm.getEmail() + " " + copiedForm.getPhone());

}

}

OUTPUT:

