1.why there is no main method in Servlets??
Answer: JVM has be designed to make a call to main method with the following prototype to start the execution

public static void main(String[] args)

If the application is standalone application then jvm will come into picture so we need main method to start the execution.

if the application is webapplication, then container will execute our java code based on its life cycle actions like

a. Loading
b. Instantiation
c. Initalization
d. RequestProcessing
e. De-Instantiation

Since jvm does not play vital role to start the execution we don't need "main method" in Servlets.

2.Eclipse Shortcuts

ctrl+2, L => to genreate the return type of method,constructor,....
Alt+ctrl+arrowup -> To Duplicate the lines.
Alt+arrowup -> To move the line upwards.
Alt+arrowdonw -> To move the line downwards.
Alt+shift+m -> To move the selected lines to be a part of the method.

Alt+shift+R -> To replace the varaible name in every place of java code.
Alt+shift+s,r => To generate setters and getters
Alt+shift+s,S => To generate toString()
Alt+shift+s,v => To do override for any methods of the class.
cntrl+shift+o => To organize the import statements/import particular class
ctrl+a => To select everything
ctrl+shift+I=> To perform indentation.
ctrl+shift+f => To perform formatting.
ctrl+shift+/ => To perform commenting.
ctrl+shift+\ => To perform uncommenting.
ctrl+shfit+t => To open particular class from the jar.
ctrl+ o => To list all the methods of the class.
ctrl+spacebar => To give assistance for our code.
ctrl+d => To delete the line
ctrl+c => To copy
ctr+v => To paste
ctrl+shift+L => To list all the shortcuts of the eclipse
ctrl+2,f => To generate a instance variable
ctrl+shift+x => to make upper case
ctrl+shift+y => to change to lower case

3.ServletConfig InitParameters Using

1.Xml
2.Annotations

a. @WebServlet(urlPatterns = {}, loadOnStartup= 10,
initParams = {
@WebInitParam(name = "url", value = "jdbc:mysql:///octbatch"),
@WebInitParam(name = "user", value = "root"),
@WebInitParam(name = "password", value = "root123")
}

Note:
	Servlet initialization parameters are key-value pair where both key and value are of type String.
	From the Servlet we can access these parameters but we can't modify.
	since we can't modify we just have only getXXXX() but not setXXXX().
	So we say Servlet Initialization parameters as "Deploy time constants".
	For every Servlet we will have only one ServletConfig object to hold its configuration information.

ServletContext(I)

==========

	For every webapplication web container will create only one ServletContext object to hold the configuration details.
	By using context object we can get configuration information like context paramters, requestdispatcher etc...

Assume there are 3 servlets and for all the servlets if the configuration details is common can we keep in config object?
	Ans. we can keep, but it is not a good practise because if we keep the data inside Context object it will be available to
		all the servlets of the application.

How to keep the data in ServletContext object?
	Ans. We can keep in only 1 way that is through XML.
		Annotation support not available becoz when container gets started only ServletContext object is created and
		no java code is coming into pitcure to give the information through Annotation.

web.xml

=======

```
<web-app>
	<context-param>
		<param-name>jdbcUrl</param-name>
		<param-value>jdbc:mysql:///octbatch</param-value>
	</context-param>
	<context-param>
		<param-name>user</param-name>
		<param-value>root</param-value>
	</context-param>
	<context-param>
		<param-name>password</param-name>
		<param-value>root123</param-value>
	</context-param>
	<servlet>
		;;;
		;;;
	</servlet>
</web-app>
```

Inside servlet we can get the ServletContext data in 2 ways
	a. ServletConfig config = getServletConfig();
	 ServletContext context= config.getServletContext();
		methods
			public String getInitParameter(String name)
			public Enumeration getInitParamterNames()

b. ServletContext context = getServletContext();
methods
public String getInitParameter(String name)
public Enumeration getInitParamterNames()

Note:
when 2 servlets have different load-on-startup then
a. lower load-on-startup will get chance first for execution

when 2 servlets have same load-on-startup then
a. it depends on container(not in the hands of the programmer)

if we give negative value for load-on-startup then the container will not load any of the servlet.
we should give only positive value (zero can also can be given)

Difference b/w ServletContext vs ServletConfig object
==========
ServletContext
a. For every webapplication, container will create only one ServletContext object to hold the data at application level.
b. It will be created at the time of application deployment and destroyed at the time of application undeployment.
c. <context-param>
<param-name></param-name>
<param-value></param-value>
</context-param>
d. 2 ways to get the Context Object
ServletContext context = getServletContext();
ServletConfig config = getServletConfig();
ServletContext context = config.getServletContext();
e. Configuration can be done only in one way through XML

ServletConfig
a. For every Servlet, container will create only seperate ServletConfig object to hold the data at servlet level.
b. It will be created at the time of Servlet object creation and destroyed at the time of Servlet Object Destruction.
c. <init-param>
<param-name></param-name>
<param-value></param-value>
</init-param>
d. Approach to get ServletConfig object
ServletConfig config = getServletConfig()
e. Configuration can be done in 2 ways
a. XML
b. Annotation