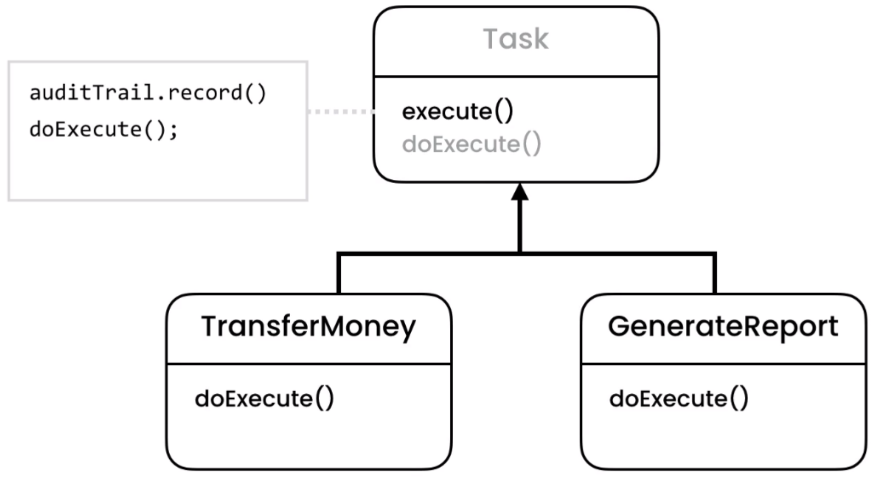
Template Method Pattern

Helps to ensure helper methods are availabe to subclasses / are always called and prevernt dupliaction of helper classes- e.g. audit trail class below or lifecycle methods when opening and closing a window.

Example Problem - Application needs to perform task e.g. transfer money and generate report while leaving an audit trail, one solution may be to have a seperate class for each task with an instance of the AuditTrail Class (calling auditTrail.record() in each task class's execute method) - issues with this include duplcate code - multiple instances of Audit Trail class and calls to it's record method, additionally it is better to program to an interface ensuring that the required methods are called

Solution - extend an abstract Task class which ensures that auditTrail.record() is always called n the execute method



Abstract Task Class	**`TransferMoneyTask - Task with Template Method pattern`**	**`GenerateReportTask - Task without pattern`**	**Audit Trail class for handling logging**

`public abstract`
`class Task {`
`  private AuditTrail`
`auditTrail;`
 	
` public Task() {`
`     auditTrail = new`
`AuditTrail();`
` }`
 	
`  public Task`
`(AuditTrail`
`auditTrail) {`
`     this.auditTrail`
`= auditTrail;`
` }`
 	
`  public void execute`
`() {`
`     auditTrail.`
`record();`
 	
` doExecute();`
` }`
 	
`  protected abstract`
`void doExecute();`
`}`
	`public class TransferMoneyTask extends Tas`
	`k {`
	`  @Override`
	`  protected void doExecute() {`
	`    System.out.println("Transfer Money");`
	` }`
	`}`
		`public class GenerateReportTask`
		`{`
		` private AuditTrail auditTrail;`
		`//required in every task`
		
		` public GenerateReportTask`
		`(AuditTrail auditTrail) {`
		`    this.auditTrail = auditTrail;`
		
		` }`
		
		`  public void execute() {`
		`  //may not be called as`
		`required`
		
		`  auditTrail.record();`
		
		` System.out.println("Generate`
		`Report");`
		` }`
		`}`
			`public class AuditTrail`
			`{`
			`  public void record()`
			`{`
			`    System.out.println(`
			`"Audit");`
			` }`
			`}`