**Code Design – Class Description**

**Classes:**

Main (Launches GUI)  
GUI (Multiple Forms) [JFrame]  
Task  
Project  
Chart (Abstract)  
Gantt (is-a Chart)  
PERT (is-a Chart)  
WBT (is-a Chart)

**Task:**

**Attributes:**

|  |  |  |
| --- | --- | --- |
| **Attribute Name** | **Data Type** | **Description** |
| name | String | The name of the task |
| Id | int | A task ID number, automatically set |
| parent | Task | The task this sub-task branches from |
| dependantNodes | Task[] | An array of all nodes that need to be done before this task |
| startDate | Date | Start date |
| endDate | Date | End date |
| lateStart | Date | Latest the task can start |
| lateEnd | Date | Latest the task can begin |
| Duration | Float | Time taken to complete the task |
| Slack | Float | The amount of time that can be left before beginning the task |
| taskNumber | String | Eg. Task “3.5” would be subtask 5 of task 3 |
| Level | int | The level of the task in the tree, (parent.level + 1), root = 0 |
| isComplete | boolean | States whether the task has been completed or not |

**Methods:**

Accessor methods for all  
Mutator methods for: name, parent, dependantNodes, startDate, endDate

Float calculateDuration();  
Float calculateSlack();  
Date calculateLateStart();  
Date calculateLateEnd();  
void complete();

**Chart:**

**Attributes:**

|  |  |  |
| --- | --- | --- |
| **Attribute Name** | **Data Type** | **Description** |
| tasks | ArrayList | An arraylist of Task[], each sub-array contains a level of tasks |
| timeFrame | Float | The total timeframe for the project |
| isValid | boolean | States whether every task has valid relationships (no isolated) |
| currentNodeCoords | Int[] | X1 y1 x2 y2 of current node. |

**Methods:**  
void drawNode(int x1, int y1, int x2, int y2);  
abstract int[] calculateNextNodeCoords();  
abstract void drawChart();  
void changeCurrentNodeCoords(int x1, int y1, int x2, int y2);

**Gantt:**

**Attributes:** all attributes inherit from Chart

**Methods:**

Void sortTasks();  
void drawNode(int x1, int y1, int x2, int y2); (Override)   
void drawChart(task currentNode, int level); (Overrides)  
Int[] getNextNodeCoords(int x, int y); (Override)

**PERT:**

**Atrributes:** all attributes inherit from Chart

**Methods:**

Void drawNode(); (Override)

Void drawLine(int startX, int endX, int startY, int endY);

Void calculateCriticalPath();

Void getNextNodeCoords(int x, int y); (Override)   
void drawChart(); (Overrides)

**WBT:**

**Attributes:** all attributes inherit from Chart

**Methods:**

Void drawLine(int startX, int endX, int startY, int endY);

Void getNextNodeCoords(int x, int y); (Override)   
void drawChart(arrayList level, task parent); (Overrides)

**Project**

**Attributes:**

|  |  |  |
| --- | --- | --- |
| **Attribute Name** | **Data Type** | **Description** |
| charts | Chart[] | Charts that are related to this project |
| name | String | Name of the project |
| filePath | String | File path of project |
| ID | Int | Unique ID of project |

Methods:

Get/sets for all attributes (except ID)

Void renameChart(int selectedChartID, String newName);

Void Load(int selectedChartID);

Void Save(Chart selectedChart);

Void delete(int selectedChartID);