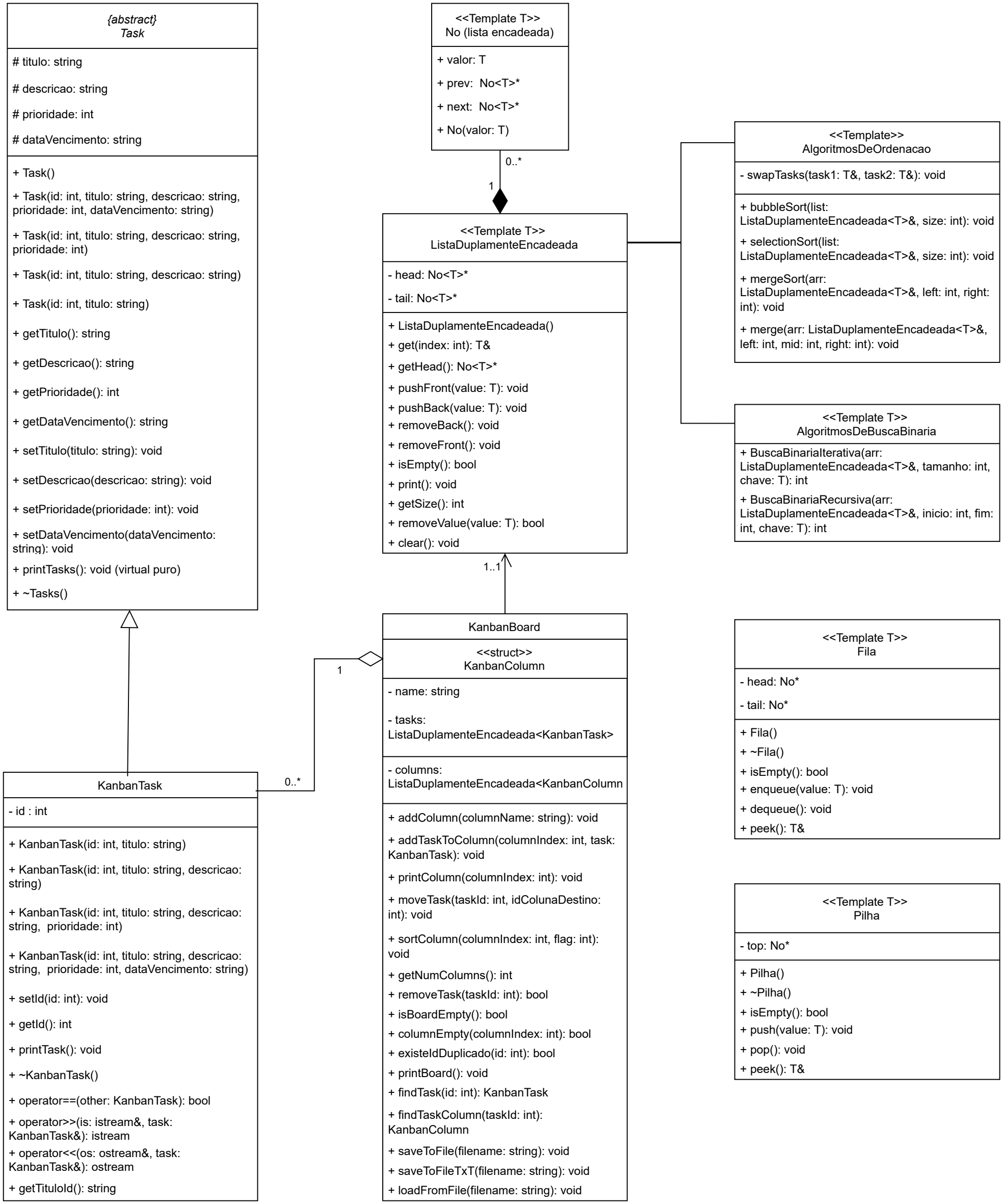
{abstract}
Task
titulo: string
descricao: string
prioridade: int
dataVencimento: string
+ Task()
+ Task(id: int, titulo: string, descricao: string, prioridade: int, dataVencimento: string)
+ Task(id: int, titulo: string, descricao: string, prioridade: int)
+ Task(id: int, titulo: string, descricao: string)
+ Task(id: int, titulo: string)
+ getTitulo(): string
+ getDescricao(): string
+ getPrioridade(): int
+ getDataVencimento(): string
+ setTitulo(titulo: string): void
+ setDescricao(descricao: string): void
+ setPrioridade(prioridade: int): void
+ setDataVencimento(dataVencimento: string): void
+ printTasks(): void (virtual puro)
+ ~Tasks()
<<Template T>>
No (lista encadeada)
+ valor: T
+ prev: No<T>*
+ next: No<T>*
+ No(valor: T)
0..*
1
<<Template T>>
ListaDuplamenteEncadeada
- head: No<T>*
- tail: No<T>*
+ ListaDuplamenteEncadeada()
+ get(index: int): T&
+ getHead(): No<T>*
+ pushFront(value: T): void
+ pushBack(value: T): void
+ removeBack(): void
+ removeFront(): void
+ isEmpty(): bool
+ print(): void
+ getSize(): int
+ removeValue(value: T): bool
+ clear(): void
<<Template>>
AlgoritmosDeOrdenacao
- swapTasks(task1: T&, task2: T&): void
+ bubbleSort(list: ListaDuplamenteEncadeada<T>&, size: int): void
+ selectionSort(list: ListaDuplamenteEncadeada<T>&, size: int): void
+ mergeSort(arr: ListaDuplamenteEncadeada<T>&, left: int, right: int): void
+ merge(arr: ListaDuplamenteEncadeada<T>&, left: int, mid: int, right: int): void
<<Template T>>
AlgoritmosDeBuscaBinaria
+ BuscaBinariaIterativa(arr: ListaDuplamenteEncadeada<T>&, tamanho: int, chave: T): int
+ BuscaBinariaRecursiva(arr: ListaDuplamenteEncadeada<T>&, inicio: int, fim: int, chave: T): int
1..1
KanbanBoard
<<struct>>
KanbanColumn
- name: string
- tasks: ListaDuplamenteEncadeada<KanbanTask>
- columns: ListaDuplamenteEncadeada<KanbanColumn
+ addColumn(columnName: string): void
+ addTaskToColumn(columnIndex: int, task: KanbanTask): void
+ printColumn(columnIndex: int): void
+ moveTask(taskId: int, idColunaDestino: int): void
+ sortColumn(columnIndex: int, flag: int): void
+ getNumColumns(): int
+ removeTask(taskId: int): bool
+ isBoardEmpty(): bool
+ columnEmpty(columnIndex: int): bool
+ existeIdDuplicado(id: int): bool
+ printBoard(): void
+ findTask(id: int): KanbanTask
+ findTaskColumn(taskId: int): KanbanColumn
+ saveToFile(filename: string): void
+ saveToFileTxT(filename: string): void
+ loadFromFile(filename: string): void
1
0..*
KanbanTask
- id : int
+ KanbanTask(id: int, titulo: string)
+ KanbanTask(id: int, titulo: string, descricao: string)
+ KanbanTask(id: int, titulo: string, descricao: string, prioridade: int)
+ KanbanTask(id: int, titulo: string, descricao: string, prioridade: int, dataVencimento: string)
+ setId(id: int): void
+ getId(): int
+ printTask(): void
+ ~KanbanTask()
+ operator==(other: KanbanTask): bool
+ operator>>(is: istream&, task: KanbanTask&): istream
+ operator<<(os: ostream&, task: KanbanTask&): ostream
+ getTituloId(): string
<<Template T>>
Fila
- head: No*
- tail: No*
+ Fila()
+ ~Fila()
+ isEmpty(): bool
+ enqueue(value: T): void
+ dequeue(): void
+ peek(): T&
<<Template T>>
Pilha
- top: No*
+ Pilha()
+ ~Pilha()
+ isEmpty(): bool
+ push(value: T): void
+ pop(): void
+ peek(): T&