

Java 基礎問題-02-02-07

Java 基礎問題-02-02-06 のクラスに

部門表 (Dept)クラスと部門レコード(RowDept)クラスを新たに追加し、
従業員表(Emp)クラス、レコード(Row)クラスは、以下クラス図を参照し新しいフィールド変数を追加してください。

従業員表(Emp)クラスのコンストラクタで、部門表 Dept を指定するように修正してください。

従業員表(Emp)にレコード(Row)を追加するとき、
Dept 表に存在しない部門番号をもつレコードは Emp 表に
挿入できないように仕様を変更してください。

部門表(Dept)の部門番号は、「0 の使用」「重複」は許されません。
部門表の中で一意に管理してください。

KadaiDept7.java に main()メソッドを作成し動作を確認してください。

Java クラスモデリング問題6の Emp
クラスに
Deptを追加

従業員表(Emp)

Row[] rowArray

Dept dept

```
void insert( Row row )  
int updateByEmpno( int empno, Row  
    updateRow )  
int deleteByEmpno( int empno )  
Row[] selectAll()  
int[] selectAllEmpno()  
Row selectByPrimaryKey( int empno )  
Row[] selectByName( String ename )  
int selectMaxSal()  
int selectRowCount()
```

レコード(Row)

```
int empno  
String ename  
String job  
int sal  
int deptno  
getter/setter
```

Java クラスモデリング問題6の Row
クラスに deptno を
追加し getter/setter
も用意する。

部門レコード(RowDept)
int deptno
String dname
getter/setter

部門表(Dept)
RowDept[] rowDeptArray
int insIndex
void insert(RowDept rowDept)
RowDept[] selectAll()

ヒント

▼必要とされる知識

オブジェクトの配列の概念

オブジェクトの参照渡し

オブジェクトの参照を返すメソッド

▼考え方のポイント

Java クラスモデリング問題 6 を使用し、効率的に実装を進めましょう。

新規追加クラスの実装内容などは no007 パッケージに格納されている Java ファイルの Javadoc コメント(`/** */`)に記載されていますので、そちらを参照し実装していきましょう。

・部門表(Dept)クラスの `selectAll()` は従業員表(Emp)クラスの `selectAll()` と同様に null を除いた有効なレコードのみを配列にまとめて返すようにします。

実行結果 ※数値と記号は半角を利用してください。

※以下実行結果の通りコンソールに出力されるように実装しましょう。
クラスモデリング問題 06 と差異がある部分を赤字にしています。

===== 部門表にレコードを追加する(insert(RowDept rowDept) メソッド) =====

部門: システム開発第1部が一行挿入されました

部門: システム開発第2部が一行挿入されました

部門番号に0は挿入できません

主キーが重なっているので挿入できません

部門: 業務推進部が一行挿入されました

部門: 業務パートナーが一行挿入されました

部門: 経理部が一行挿入されました

もう挿入できません

===== 部門表にレコードを追加する(insert(RowDept rowDept) メソッド) =====

===== 従業員表にレコードを追加する(insert(Row row) メソッド) =====

すみすさんの情報が一行挿入されました

ありすさんの情報が一行挿入されました

部門番号が部門表に存在しません

主キーが重なっているので挿入できません

従業員番号に0は挿入できません

けんさんの情報が一行挿入されました

めありーさんの情報が一行挿入されました

けんさんの情報が一行挿入されました

もう挿入できません

===== 従業員表にレコードを追加しました(insert(Row row) メソッド) =====

===== 今の従業員表登録件数を表示開始(selectRowCount() メソッド) =====

今従業員表に35件登録されています。

===== 今の従業員表登録件数を表示終了(selectRowCount() メソッド) =====

===== 指定した従業員番号に該当するレコードを削除開始(deleteByEmpno(int empno) メソッド) =====

削除された行は0

削除された行は0

===== 指定した従業員番号に該当するレコードを削除終了(deleteByEmpno(int empno) メソッド) =====

===== 今の従業員表登録件数を表示開始(selectRowCount() メソッド) =====

今従業員表には4件登録されています。

===== 今の従業員表登録件数を表示終了(selectRowCount() メソッド) =====

===== 指定した従業員番号に該当するレコードを更新開始(updateByEmpno(int empno, Row updateRow) メソッド) =====

更新された行は0

更新行数は0

===== 指定した従業員番号に該当するレコードを更新終了(updateByEmpno(int empno, Row updateRow) メソッド) =====

===== 従業員名で検索開始(selectByName(String ename) メソッド) =====

従業員番号:3333, 名前: けん, 役職: 一般社員, 部門番号:2

従業員番号:5555, 名前: けん, 役職: 派遣社員, 部門番号:4

===== 従業員名で検索終了(selectByName(String ename) メソッド) =====

===== 従業員表に登録されている従業員中、最も高い給料検索開始(selectMaxSal() メソッド) =====

一番高い給料は8400

===== 従業員表に登録されている従業員中、最も高い給料検索終了(selectMaxSal() メソッド) =====

===== 指定した従業員番号に当てはまるレコード検索開始(selectByPrimaryKey(int empno) メソッド) =====

従業員番号:3333, 名前: けん, 役職: 一般社員, 部門番号:2

===== 指定した従業員番号に当てはまるレコード検索終了(selectByPrimaryKey(int empno) メソッド) =====

===== 従業員表に登録されている全レコード取得開始(selectAll() メソッド) =====

従業員番号:1111, 名前: すみず, 役職: 一般社員, 部門番号:1

従業員番号:2222, 名前: NEWありす, 役職: 受付係, 部門番号:3

従業員番号:3333, 名前: けん, 役職: 一般社員, 部門番号:2

従業員番号:5555, 名前: けん, 役職: 派遣社員, 部門番号:4

===== 従業員表に登録されている全レコード取得終了(selectAll() メソッド) =====

===== 従業員表に登録されている全レコードの従業員番号一覧取得開始(selectAllEmpno() メソッド) =====

1111

2222

3333

5555

===== 従業員表に登録されている全レコードの従業員番号一覧取得終了(selectAllEmpno() メソッド) =====