# Recursive User-Defined Algebraic Data Types

## import_list

CS 421

Revision 1.0

## 1 Change Log

**1.0** Initial Release.

## 2 Objectives

Your objectives are:

- Constructing data structures from algebraic data types

1. Using the type `exp` found in `common.ml` and described in `cps-adt-info.pdf`, write a function `import_list : (int * int) list -> exp`, that takes a list of pairs and converts it into an expression in our language that is equivalent to it.

```
# let rec import_list lst = ...;;
val import_list : (int * int) list -> Common.exp = <fun>
# import_list [(7,1);(4,2);(6,3)];;
- : Common.exp =
BinOpAppExp (ConsOp,
 BinOpAppExp (CommaOp, ConstExp (IntConst 7), ConstExp (IntConst 1)),
 BinOpAppExp (ConsOp,
  BinOpAppExp (CommaOp, ConstExp (IntConst 4), ConstExp (IntConst 2)),
  BinOpAppExp (ConsOp,
   BinOpAppExp (CommaOp, ConstExp (IntConst 6), ConstExp (IntConst 3)),
   ConstExp NilConst)))
```