2005 Paper 13 Question 10

Introduction to Functional Programming

(*a*) Define a function `subsets` in ML with the type

```
subsets :  'a list -> 'a list list
```

which treats its input list as a set of n elements and returns a list of all 2^n subsets of that set. [5 marks]

(*b*) Define an exception `NoFit` with no arguments, and an exception `Success` whose constructor takes an `int list` as an argument. [1 mark each]

(*c*) Use `subsets` to define a function `knapsack` with the type

```
knapsack :  int -> int list -> int list
```

which finds a subset of the `int`s in its second argument whose sum is exactly the first argument. Use at least one functional (`map`, `foldl`, etc.) in the definition of `knapsack`. The `NoFit` exception should be raised in the event that no solution exists. [5 marks]

(*d*) Define another version of `knapsack` called `knapsack2` with the type

```
knapsack2 :  int -> int list -> unit
```

`knapsack2` should use simple recursion to generate its candidate solutions on-the-fly as it tests them, and should raise `Success` with the solution on success or return `unit` if no solution exists. [8 marks]