One way to represent sets of integers is as lists of intervals:

```ocaml
type intset = (int * int) list
```

For example, \{1, 2, 3, 9, 10, 11, 12\} can be represented as \([(1, 3); (9, 12)]\), the union of the intervals [1..3] and [9..12].

(a) Each set of integers has many different interval list representations. An interval list (intset) is in standard form if it is an ascending sequence of non-empty intervals that cannot be merged.

(i) Write a function that tests whether an intset is in standard form:

```ocaml
val is_standard : intset -> bool
```

[4 marks]

(ii) Write a function that adds an interval to an intset in standard form, producing a new intset in standard form:

```ocaml
val add_interval : (int * int) -> intset -> intset
```

[4 marks]

(iii) Write a function that converts an intset to standard form:

```ocaml
val standardize : intset -> intset
```

[2 marks]

(iv) Write a function to test whether two intset values represent the same set:

```ocaml
val equal : intset -> intset -> bool
```

[2 marks]

(b) Write a function that computes the intersection of integer sets:

```ocaml
val inter : intset -> intset -> intset
```

You may assume that the arguments to inter are in standard form. [8 marks]