CPSC326 | Homework 8: O' Camel+ | 5/3/2025 | Arjuna Herbst

Tests for each function:

```
# eval (If (Int 0, Int 10, Int 20));;
- : int = 20
# eval (Iterate (3, (fun x -> x + 1), Int 2));;
- : int = 5
# eval (Iterate (4, (fun x -> x * 2), Int 1));;
- : int = 16
```

Tests for each of the difficult expressions

```
# let s = Elem (1, Elem (3, EmptySet));;
val s : int set = Elem (1, Elem (3, EmptySet))
# add 2 s;;
- : int set = Elem (1, Elem (2, Elem (3, EmptySet)))
```

Adding a element to a set

```
# let s1 = Elem (1, Elem (3, Elem (5, EmptySet)));;
val s1 : int set = Elem (1, Elem (3, Elem (5, EmptySet)))
# member 1 s1;;
- : bool = true
# member 8 s1;;
- : bool = false
```

Element in the list vs not in the list

```
let s1 = Elem (1, Elem (3, Elem (5, EmptySet)));;
let s2 = Elem (1, Elem (2, Elem (3, Elem (4, Elem (5, EmptySet)))));;
```

```
# subset s1 s2;;
- : bool = true
# subset s2 s1;;
- : bool = false
```

s1 is a subset of s2, but not vice versa

```
let s1 = Elem (1, Elem (3, Elem (5, EmptySet)));;
let s2 = Elem (1, Elem (3, Elem (5, EmptySet)));;
```

```
# equal s1 s2;;
- : bool = true
```

s1 and s2 are equal

```
# let s3 = Elem (2, Elem (3, Elem (4, EmptySet)));;
val s3 : int set = Elem (2, Elem (3, Elem (4, EmptySet)))
# let s1 = Elem (1, Elem (3, Elem (5, EmptySet)));;
val s1 : int set = Elem (1, Elem (3, Elem (5, EmptySet)))
# union s1 s3;;
- : int set = Elem (1, Elem (2, Elem (3, Elem (4, Elem (5, EmptySet)))))
```

s3 ∪ s1

```
# let s1 = Elem (1, Elem (3, Elem (5, EmptySet)));;
val s1 : int set = Elem (1, Elem (3, Elem (5, EmptySet)))
# let s2 = Elem (1, Elem (2, Elem (3, Elem (4, Elem (5, EmptySet)))));;
val s2 : int set = Elem (1, Elem (2, Elem (3, Elem (4, Elem (5, EmptySet)))))
# intersect s1 s2;;
- : int set = Elem (1, Elem (3, Elem (5, EmptySet)))
```

Find the elements that both s1 and s2 have in common

```
# let s1 = Elem (1, Elem (3, Elem (5, EmptySet)));;
val s1 : int set = Elem (1, Elem (3, Elem (5, EmptySet)))
#
  set_map (fun x -> x * 2) s1;;
- : int set = Elem (2, Elem (6, Elem (10, EmptySet)))
```

Multiply elements in the set by 2

```
# set_map (fun x -> x mod 2) s1;;
- : int set = Elem (1, EmptySet)
```

Mod each element in set by 2