Lista 4

Algebraiczne typy danych

W poniższych zadaniach **dopuszczalne jest** wykorzystanie funkcji wbudowanych obliczających **długość listy**, **odwracających listę** oraz **łączących dwie listy**, o ile **nie wpływają one na drastyczne pogorszenie złożoności obliczeniowej**.

Każde zadanie, poza implementacją funkcji, musi posiadać **kompletny zestaw testów**.

Do wykonania zadań należy wykorzystać mechanizmy poznane na wykładzie nr 4.

W języku OCaml definicje algebraicznych typów danych zazwyczaj zawierają krotki, np.:

```
type 'a lst = Nil | Cons of 'a * 'a lst;;
```

Pozwala to na tworzenie wartości postaci:

```
let xs = Cons(1, Cons(2, Nil));;
```

Okazuje się jednak, że definicja typu może zawierać rekordy (typy rekordowe), np.:

```
type 'a lst = Nil | Cons of { head : 'a; tail : 'a lst };;
```

Taka definicja pozwala na nazywanie składowych typu np.:

```
let xs = Cons{ head = 1; tail = Cons{ head = 2; tail = Nil} };;
```

W tej postaci, po dopasowaniu do wzorca, można operować na polach rekordu po nazwie:

```
let Cons elem = xs in elem.tail;;
```

W poniższych zadaniach, dla języka OCaml należy wykorzystać algebraiczne typy danych z rekordami!

1) Zdefiniuj:
   a. Typ *tree3* (drzewo trójkowe) o następujących cechach: (OCaml i Scala) (5 pkt.)
      - Drzewo puste jest drzewem trójkowym,
      - Węzeł drzewa składa się z elementu oraz 0 – 3 poddrzew.

   b. Zdefiniuj funkcjonał *combineTree3* w postaci rozwiniętej, łączący dwa drzewa trójkowe za pomocą funkcji. Łączenie polega na przejściu drzew taki sam sposób i przekazanie do funkcji wartości z obu węzłów. Jeśli dany węzeł nie ma swojego odpowiednika, to w drzewie wynikowym ma zostać wstawiony węzeł pusty. (OCaml lub Scala) (15 pkt.)

2) Zapoznaj się z poniższym opisem wycinka rzeczywistości i wykonaj zadania:

„Dane są przechowywane na dysku. Dysk jest identyfikowany przez literę i może zawierać pliki i foldery. Każdy folder posiada nazwę i może przechowywać pliki i foldery. Plik posiada nazwę, nie może posiadać podfolderów ani plików.”

a. Bazując na powyższym opisie zdefiniuj algebraiczny typ danych modelujący wycinek rzeczywistości. Pamiętaj, że typy często reprezentują kluczowe byty. Zastosuj angielskie nazwy. (OCaml i Scala) (15 pkt.)

b. Zdefiniuj funkcję *path*, która dla danego systemu plików (typy zdefiniowane w podpunkcie a) oraz nazwy zwraca ścieżkę do zasobu – folderu lub pliku. Zwracana ścieżka niech będzie reprezentowana przez łańcuch znaków nazw elementów systemu plików od korzenia do katalogu zawierającego zasób np. „C:\Program Files\Microsoft Office\”. W przypadku występowania zasobu w wielu miejscach struktury zwrócić lokalizację najbliższą korzenia drzewa. Brak zasobu wyrazić za pomocą typu opcjonalnego. (OCaml) (15 pkt.)