1. Pentru a instantia clasa Eq trebuie sa implementam urmatoarele functii:
a) ==, /=, >, >=
b) eq, not
c) ==
d) eq
2. Ce constrangeri de tipuri trebuie sa adaugam la functia f pentru ca urmatorul cod sa fie corect?
data MyData a b = MyData a b b
f :: MyData a b -> MyData a b -> Bool
f (MyData x1 y1 z1) (MyData x2 y2 z2) = x1 == x2 && y1 == y2 && z1 == z2
a) Eq a, Eq b
b) Eq a
c) Eq a, Ord a
d) codul este deja corect;
3. Se defineste:
data MyData a b = MyData a b b
Care instanta de mai jos este corecta?
a) instance Functor (MyData a b) where
fmap f (MyData x y z) = MyData x (f y) (f z)
b) instance Functor (MyData a) where
fmap f (MyData x y z) = MyData (f x) (f y) (f z)
c) instance Functor (MyData a) where
fmap f (MyData x y z) = MyData x (f y) (f z)
d) instance Functor (MyData a b) where
fmap f (MyData x y z) = MyData (f x) (f y) (f z)
4. Se defineste
data MyData a b = Data1 a | Data2 b
Care instanta de mai jos este corecta?
a) instance Functor (MyData a b) where
fmap f (Data1 x) = Data1 x
fmap f (Data2 x) = f (Data2 x)
b) instance Functor (MyData a) where
fmap f (Data1 x) = Data1 (f x)
fmap f (Data2 x) = Data2 (f x)
c) instance Functor (MyData a) where
fmap f (Data1 x) = Data1 x
fmap f (Data2 x) = Data2 (f x)
d) instance Functor (MyData a b) where
fmap f (Data1 x) = Data1 (f x)
fmap f (Data2 x) = Data2 (f x)
5. Care instanta Monoid de mai jos este corecta?
newtype MyBool = MyBool Bool
deriving (Eq, Show)
a) instance Monoid MyBool where
MyBool x <> MyBool y = MyBool (x && y)
mempty = MyBool True
b) nu se poate face instanta Monoid pentru tipul MyBool.
c) instance Semigroup MyBool where
MyBool x <> MyBool y = MyBool (x || y)
instance Monoid MyBool where
mempty = MyBool True

d) instance Semigroup MyBool where
MyBool x <> MyBool y = MyBool (x && y)
instance Monoid MyBool where
mempty = MyBool True
6. Care instanta Monoid de mai jos este corecta?
newtype MyInt = MyInt Int
deriving (Eq, Show)
a) instance Semigroup MyInt where
MyInt x <> MyInt y = MyInt (x + y + 1)
instance Monoid MyInt where
mempty = MyInt 0
b) nu se poate face instanta Monoid pentru tipul MyInt
c) instance Semigroup MyInt where
MyInt x <> MyInt y = MyInt (x + y + 1)
instance Monoid MyInt where
mempty = MyInt (-1)
d) instance Semigroup MyInt where
MyInt x <> MyInt y = MyInt (x + y + 1)
mempty = MyInt (-1)
7. Se defineste:
data MyData a b = MyData a b b
Care instanta de mai jos este corecta?
a) instance Foldable (MyData a) where
foldMap f (MyData x y z) = f z
b) instance Foldable (MyData a b) where
foldMap f (MyData x y z) = f y <> f z
c) instance Foldable (MyData a) where
foldMap f (MyData x y z) = f y <> f z
d) instance Foldable (MyData a) where
foldMap f (MyData x y z) = f x <> f y <> f z
8. Se defineste:
data MyData a b = Data1 a | Data2 b
Care instanta de mai jos este corecta?
a) instance Foldable (MyData a b) where
foldMap f (Data1 x) = Data1 x
foldMap f (Data2 x) = f x
b) instance Foldable (MyData a) where
foldMap f (Data1 x) = f x
foldMap f (Data2 x) = f x
c) instance Foldable (MyData a) where
foldMap f (Data1 x) = mempty
foldMap f (Data2 x) = f x
d) instance Foldable (MyData a) where
foldMap f (Data1 x) = Data1 x
foldMap f (Data2 x) = f x
9. Ce se obtine dupa instructiunea [(+ 1) , (^2)] <*> [1 , 2 , 3, 4]?
a) [2,3,4,5,1,4,9,16]
b) [2,3,4,5]
c) instructiune invalida
d) [1,4,9,16]
10. Ce se obtine dupa instructiunea (+10) <*> [1..5]?

a) [11, 12,13,14,15]
b) [10,20,30,40,50]
c) instructiune invalida
d) [1,2,3,4,5]

-- grile
-- 1 c
-- 2 a
-- 3 c
-- 4 c
-- 5 d
-- 6 c
-- 7 c
-- 8 c
-- 9 a
-- 10 c