Test PP Numele ..... Grupa .....

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
a																				
b																				
c																				
d																				

1. Fie următoarele definiții Haskell:

```
f1 = filter odd . map (+1)
f2 = map (+1) . filter even
f3 = filter odd $ map (+1)
f4 = map (+1) $ filter even
```

Care din următoarele afirmații este adevarată?

(a) `f1 [2,5..10] == f2 [2,5..10]`
(b) `f1 [2,5..10] == f3 [2,5..10]` și `f2 [2,5..10] == f4 [2,5..10]`
(c) `f1 [2,5..10] == f2 [3,6..11]` și `f3 [2,5..10] == f4 [3,6..11]`
(d) `f1 [2,5..10] == f2 [1,4..9]` și `f3 [2,5..10] == f4 [1,4..9]`

2. Care dintre următoarele funcții Haskell păstrează valorile `True` dintr-o listă de `Bool`?

(a) `filter id`
(b) `filter not`
(c) `map id`
(d) `map not`

3. Ce va afișa următorul program?

```
(define (f x) (* x 2))
(define L '(1 2 3 4))
(foldl (lambda (elem acc) (cons (f elem) acc)) '() L)
```

(a) `(8 6 4 2)`
(b) `(2 4 6 8)`
(c) `()`
(d) eroare

4. La ce se va evalua următoarea expresie?

```
(filter list? (cons 0 '(1 2 3)))
```

(a) `()`
(b) `(0 1 2 3)`
(c) `(1 2 3)`
(d) `(#f #f #t)`

5. Care este valoarea expresiei `let` exterioare?

```
(define x 0)
(let ([x 1])
  (let ([x 2]
        [y x])
    y))
```

(a) 1
(b) 0
(c) 2
(d) Eroare

6. Ce va afișa următorul program Racket?

```
(define f (lambda (f) (f f)))
(define g (f (lambda (f) f)))
(equal? (g g) g)
```

(a) `#t`
(b) `#f`
(c) Programul nu se va termina
(d) Eroare

7. Ce va întoarce următorul cod Racket:

```
(define f (lambda () f))
(f)
```

(a) O funcţie
(b) O eroare
(c) O promisiune
(d) Programul intră în buclă infinită

8. Care dintre următoarele expresii va produce eroare în Racket?

(a) `(or #f (/ 1 0) #t)`
(b) `(lambda (x) (car cdr))`
(c) `(lambda () (/ 1 0))`
(d) `(delay (/ 1 0))`

9. Care este tipul următoarei funcții Haskell?

```
f x = [f $ x + 1]
```

(a) Eroare de sinteză
(b) `a -> [a]`
(c) `a -> [b]`
(d) `Num a => a -> [a]`

10. Ce tip are următoarea funcţie în Haskell?

```
f x y = head [x, y, f x y]
```

(a) `a -> a -> a`
(b) `a -> b -> a`
(c) `a -> b -> c`
(d) `a -> b -> [a]`

11. Care din următoarele expresii Haskell va genera o eroare?

(a) `'a' ++ "bcd"`
(b) `['a'] ++ "bcd"`
(c) `"a" ++ "bcd"`
(d) `'a' : "bcd"`

12. Fie următorul program Haskell:

```
data Expr
  =  Atom Int
  | Plus Expr Expr
  | Mult Expr Expr
  deriving (Show, Eq)
eval (Atom x)   = x
eval (Plus x y) = eval x + eval y
eval (Mult x y) = eval x * eval y
```

Care va fi rezultatul evaluării expresiei:

```
eval $ Mult (Mult (Plus (Atom 2) (Atom 3)) (Atom 5))
            (Plus (Mult (Atom 10) (Atom 1)) (Atom 0))
```

(a) 250
(b) 0
(c) Evaluarea va genera eroare de tip
(d) 42

13. Ce fel de polimorfism utilizează următoarea funcție în Haskell?

```
f :: Ord a => a -> a -> b -> b -> b
f x y a b = if x < y then a else b
```

(a) Atât parametric, cât și ad-hoc
(b) Exclusiv parametric
(c) Exclusiv ad-hoc
(d) Funcția nu este polimorfică

14. Fie următorul program în Haskell:

```
f x = x + x
g x = 2 * x
```

Presupunând că funcția `(==) :: Eq a => a -> a -> Bool` este importată implicit la încărcarea în interpretor, ce rezultat va întoarce expresia f == g?

(a) Eroare de tip
(b) Eroare de sintaxă
(c) `True`
(d) `False`

15. Fie următoarele fapte în cadrul unui program Prolog:

```
p(1). p(2). p(4).
q(1). q(4). q(8).
```

Ce va afisa următoarea interogare?

```
setof(X/Y, (p(X), q(Y), not(q(X))), L).
```

(a) `L = [2/1, 2/4, 2/8].`
(b) `L = [2.0, 0.5, 0.25].`
(c) `L = [1/4, 1/8, 2/1, 2/4, 2/8, 4/1, 4/8].`
(d) `L = [0.25, 0.125, 2.0, 0.5, 4.0].`

16. Ce întoarce următoarea interogare Prolog?

```
append([_,_|_], L, [1,2,3]).
```

(a) `L = [3]; L = []; false.`
(b) `Error: Arguments not sufficiently instantiated`
(c) `L = [1,2,3]; L = [2,3]; L = [3]; false.`
(d) `L = []; L = [1]; false.`

17. Ce va afișa interogarea `member(X, [Y, Z]).` în Prolog?

(a) `X = Y; X = Z.`
(b) `false.`
(c) `true.`
(d) `Error: Arguments not sufficiently instantiated`

18. Ce valoare va avea `Z` în urma rulării în linia de comanda a următoarei comenzi Prolog?

```
?- X = Y, Y = 5-3, Z = X mod 2.
```

(a) `(5-3) mod 2`
(b) `0`
(c) Eroare
(d) `2 mod 2`

19. Se dă următorul program în Prolog:

```
p(X) :- !, q(X), r(X).
q(1).
q(2).
r(1).
r(2).
```

Ce va întoarce încercarea de a satisface scopul `p(X)`?

(a) `X = 1; X = 2.`
(b) `X = 1.`
(c) `X = 2.`
(d) `false.`

20. Fie următoarele definiții în Racket, respectiv Haskell:

```
(define f (lambda (x y) x))
f x y = x
```

La ce se va evalua expresia `(f 1)` în Racket, respectiv Haskell?

(a) Eroare, o funcție
(b) O funcție, eroare
(c) O funcție, o funcție
(d) Eroare, eroare