# Programming Assignment on Primitive Procedures & Compound Procedures

**Predict the output for below LISP expressions.**

1. **Primitive Expressions**
   - 500
   - 300
   - 100
   - -300
   - 10
   - 2/5
   - 5/2
   - 2.5
   - 2.5
   - (5
   - 25
   - 1.41421
   - 0
   - 2
   - 8
   - 5
   - **Working with strings**
   - "Welcome to LISP Programming"
   - Hi
2. How are you
   - **Few Exceptions**
   - 0
   - 1
   - 1
   - 0
   - 1
   - #t
   - **Working with Boolean and Relational Expressions**
   - - Error
   - - #t
   - - #t
   - - #f
   - - #t
   - - #t
   - - #f
   - - #f
   - - #t
   - - #t
   - - #f

   - **Built-in Procedures**
     - 0
     - 1
     - 8
     - 3

Combination / Compound Expressions / Nested Expressions

- 10
- 6
- -2
- 20
- 32
- 57

3. **Naming and Environment (define variables and assign values/expressions)**

i)

➢ 25

ii)

➢ 10

iii)

➢ 8

iv)

➢ 7

v)

➢ 314.0

vi)

➢ "Hi"

4. **"define" as simplest form of abstraction / Simple procedures and Substitution**

i)

Circumference:

62.800000000000004

Area:

314.0

ii)

200

iii)

3140.0

**Programs on Procedures: -**

1. Define the following procedures, and compute them as mentioned: -

```
Welcome to Racket v7.2.
> (define (square x) (* x x))
> (square 5)
25
> (define (cube x) (* x x x))
> (cube 5)
125
> (define (cube2 x) (* (square x) x))
> (cube2 5)
125
> (define (power1 x) (* (cube x) x))
> (power1 5)
625
> (define (power2 x) (* (cube2 x) x))
> (power2 5)
625
> (define (power3 x) (* (square x) (square x)))
> (power3 5)
625
> (define (power4 x) (* (square (square x))))
> (power4 5)
625
>
```

2. Define a procedure “square” to compute square of a number. Define another procedure “sum-of-squares” that calculates sum of squares of two numbers using the procedure “square”. Using these procedures, define the procedures to compute the following.

```
Welcome to Racket v7.2.
> (define (square x) (* x x))
> (define (sum_of_squares x y) (+ (square x) (square y)))
> (square 5)
25
> (sum_of_squares 3 4)
25
> (define (area r) (* 3.14 (square r)))
> (area 10)
314.0
> (define (hypotnuse a b) (sqrt (sum_of_squares a b)))
> (hypotnuse 3 4)
5
> (define (dist a b c d) (sqrt (sum_of_squares (- a c) (- b d) ))
  )
> (dist 0 0 1 1 )
1.4142135623730951
> (dist -7 -4 17 6.5 )
26.196373794859472
>
```

*******