# STAT 40001/ STAT 50001 Statistical Computing Fall 2024

**Lab-1**

**1.** Calculate the following

a) 45÷5+49×67-98

```
> 45/5+49*67-98
[1] 3194
```

b) 4+(35mod 6)+9 ln(5)

```
> 4+(35%%6)+9*log(5)
[1] 23.48494
```

c) 67÷5+9×37

```
> 67/5+9*37
[1] 346.4
```

d) |-7|+|5|+log(10)

```
> abs(-7)+abs(5)+log(10)
[1] 14.30259
```

e) $\sqrt{49}+67+\sqrt{873}$

```
> sqrt(49)+67+sqrt(873)
[1] 103.5466
```

f) $78+\ln(45)+e^{7}$

```
> 78+log(45)+exp(7)
[1] 1178.44
```

**2.** Generate sequence of even numbers between 10 and 50.

```
> even_numbers_sequence = seq(from=10, to=50, by=2)
> even_numbers_sequence
 [1] 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48
50
```

**3.** Generate a sequence of 100 numbers between 1 and 100.

a) Print only the first 5 numbers.

```
> sequence = seq(from=1,to=100)
> req_sequence = head(sequence,5)
> req_sequence
[1] 1 2 3 4 5
```

b) Print only the last 5 numbers.

```
> sequence = seq(from=1,to=100)
> req_sequence = tail(sequence,5)
> req_sequence
[1]  96  97  98  99 100
```

**4.** We have several way of rounding the numbers including

`ceiling` takes a single numeric argument `x` and returns a numeric vector containing the smallest integers not less than the corresponding elements of `x`.

`floor` takes a single numeric argument `x` and returns a numeric vector containing the largest integers not greater than the corresponding elements of `x`.

`round` rounds the values in its first argument to the specified number of decimal places (default 0).

Given three numbers 1.023456, 5.45768, and  1.678927 use the following options

i) round

```
> x
[1] 1.023456 5.457680 1.678927
> round(x)
[1] 1 5 2
> round(x,2)
[1] 1.02 5.46 1.68
```

ii) ceiling

```
> x
[1] 1.023456 5.457680 1.678927
> ceiling(x)
[1] 2 6 2
```

iii) floor

```
> x
[1] 1.023456 5.457680 1.678927
> floor(x)
[1] 1 5 1
```

**5.** Sort the data in decreasing order:

3, 5, 7, 2, 9, 12, 45, 23, 31, 45, 7, 82, 90, 5

```
> x = c(3, 5, 7, 2, 9, 12, 45, 23, 31, 45, 7, 82, 90,5)
> x
 [1]  3  5  7  2  9 12 45 23 31 45  7 82 90  5
```

```
> sort(x, decreasing=T)
 [1] 90 82 45 45 31 23 12  9  7  7  5  5  3  2
```

**6.** Sort the data in increasing order

4,6,7,8,2,3,6,8,4,9,15,34,23,81,-5,-9

```
> x
 [1]  4  6  7  8  2  3  6  8  4  9 15 34 23 81 -5 -9
> sort(x)
 [1] -9 -5  2  3  4  4  6  6  7  8  8  9 15 23 34 81
```