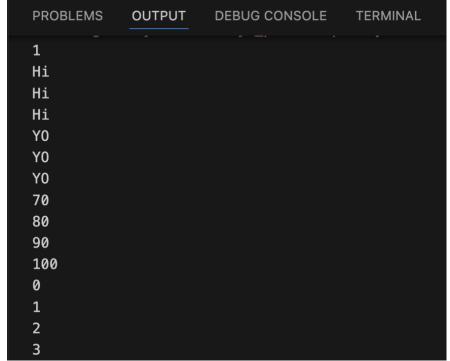
Csc600- Homework Ruby

Name – Mya Phyu

1)



```
4
5
6
7
8
9
5
10
15
20
25
30
35
40
45
```

```
50
0
1
2
3
4
5
6
4
5
6
7
7
7
6
5
4
0
10
20
30
40
50
60
Welcome
to
Railway
Portal
2
4
6
collect a: [2. 3. 4. 5]
```

select method : [18, 22, 50] reject method : [3, 3, 6]

```
2)
         class Array
             def limited?(amin, amax)
             self.all? { |x| \times = amin \&\& \times <= amax }
             def sorted?
              return 0 if self.empty?
               increasing = self.each_cons(2).all? { |a, b| a <= b }</pre>
               decreasing = self.each_cons(2).all? { |a, b| a >= b }
               return 1 if increasing
               return -1 if decreasing
               0
         end
         arr1 = [1, 2, 3, 4, 5]
         arr2 = [5, 4, 3, 2, 1]
  108
         arr3 = [3, 2, 5, 4, 1]
         arr4 = [10, 20, 30, 40]
         puts arr1.limited?(1, 5)
         puts arr2.limited?(1, 5)
         puts arr3.limited?(1, 5)
         puts arr4.limited?(10, 40)
         puts arr1.sorted?
         puts arr2.sorted?
         puts arr3.sorted?
         puts arr4.sorted?
  120 _
     true
     true
     true
     true
     1
     -1
```

0

1

```
class Triangle
    attr_accessor :a, :b, :c
    def initialize(a, b, c)
     @a = a
     @b = b
     @c = c
    def perimeter
    return @a + @b + @c
    def area
     s = self.perimeter / 2.0
     Math.sqrt(s * (s - @a) * (s - @b) * (s - @c))
    def test
     sides = [@a, @b, @c].sort
     if sides[0] + sides[1] \le sides[2]
        puts "Not a triangle"
     elsif @a == @b && @b == @c
       puts "Equilateral Triangle"
     elsif @a == @b || @b == @c || @c == @a
        puts "Isosceles Triangle"
      elsif @a**2 + @b**2 == @c**2 || @a**2 + @c**2 == @b**2 || @b**2 + @c**2 == @a**2
       puts "Right Triangle"
       puts "Scalene Triangle"
triangle = Triangle.new(3, 4, 5)
triangle.test
puts "Perimeter: #{triangle.perimeter}"
puts "Area: #{triangle.area}"
```

Right Triangle

Perimeter: 12

Area: 6.0

```
class Sphere
   attr_accessor :radius
   def initialize(radius)
    @radius = radius
    def area
    4 * Math::PI * @radius**2
    def volume
    (4.0 / 3.0) * Math::PI * @radius**3
end
  class Ball < Sphere</pre>
    attr_accessor :color
    def initialize(radius, color)
     super(radius)
     @color = color
  class MyBall < Ball</pre>
    attr_accessor :owner
    def initialize(radius, color, owner)
     super(radius, color)
     @owner = owner
    def show
      puts "Owner: #{@owner}"
      puts "Color: #{@color}"
     puts "Radius: #{@radius}"
puts "Surface Area: #{area}"
      puts "Volume: #{volume}"
my_ball = MyBall.new(5, "Red", "John")
my_ball.show
```

Owner: John

Color: Red

Radius: 5

Surface Area: 314.1592653589793

Volume: 523.5987755982989