1.Create a function that takes in a name as a string argument, and prints out "Hello name".

Input:

```
hello_you <- function(name){
  print(paste('Hello',name))
}
```

Input:

```
hello_you('Sam')
```

```
hello_you <- function(name='Sam'){
  print(paste("Hello",name))
}

hello_you('Sam')
```

2.Create a function that takes in a name as a string argument and *returns* a string of the form - "Hello name"

Input: print(hello_you2('Sam'))

```
hello_you2 <- function(name='Sam'){
  return(list("Hello",name))
}

print(hello_you2('Sam'))
```

3.Create a function that will return the product of two integers.

Input: prod(3,4)
Output: 12

```
prod <- function(num1,num2){
```

```
  my.mul <- num1*num2
  return(my.mul)
}
result <- prod(3,4)
result
```

4.Create a function that accepts two arguments, an integer and a vector of integers. It returns TRUE if the integer is present in the vector, otherwise it returns FALSE. Make sure you pay careful attention to your placement of the return(FALSE) line in your function!

Input: num_check(2,c(1,2,3))
Output: TRUE
Input: num_check(2,c(1,4,5))
Output: FALSE

```
v <- c
num_check <- function(num1,v){
  for (item in v) {
    if (item == num1){
      return (TRUE)
    }
  }
      return (FALSE)
}

num_check(2,c(1,2,3))
```

5.Create a function that accepts two arguments, an integer and a vector of integers. It returns the count of the number of occurrences of the integer in the input vector.

Input: num_count(2,c(1,1,2,2,3,3))

Output:2
Input: num_count(1,c(1,1,2,2,3,1,4,5,5,2,2,1,3))
Output:4

```
v <- c
count <- function(num,v){
  count = 0
  for (item in v) {
    if (item == num){
      count = count + 1
    }
  }
  return (count)
}
count(2,c(1,1,2,2,3,3))
count(1,c(1,1,2,2,3,1,4,5,5,2,2,1,3))
```

6.We want to ship bars of aluminum. We will create a function that accepts an integer representing the requested kilograms of aluminum for the package to be shipped. To fullfill these order, we have small bars (1 kilogram each) and big bars (5 kilograms each). Return the least number of bars needed.
For example, a load of 6 kg requires a minimum of two bars (1 5kg bars and 1 1kg bars). A load of 17 kg requires a minimum of 5 bars (3 5kg bars and 2 1kg bars).

Input: bar_count(6)
Output: 2
Input: bar_count(17)
Output:5

```
bar_count <- function(load){
  amount1 <- load %% 5
  amount2 <- (load - amount1)/5
```

```
  return (amount1 + amount2)
}


bar_count(6)
bar_count(17)
```

7.Create a function that accepts 3 integer values and returns their sum. However, if an integer value is evenly divisible by 3, then it does not count towards the sum. Return zero if all numbers are evenly divisible by 3. Hint: You may want to use the append() function.

Input: summer(7,2,3)
Output:9
Input: summer(3,6,9)
Output:0
Input: summer(9,11,12)
Output:11

```
summer <- function(n1,n2,n3){
  out <-c(0)
  if(n1 %% 3 != 0){
    out <- append(n1,out)
  }
  if(n2 %% 3 != 0){
    out <- append(n2,out)
  }
  if(n3 %% 3 != 0){
    out <- append(n3,out)
  }
  return(sum(out))
}
summer(7,2,3)
```

```
summer(3,6,9)


summer(9,11,12)
```

8.Create a function that will return TRUE if an input integer is prime. Otherwise, return FALSE. You may want to look into the any() function.

Input: prime_check(2)
Output:TRUE
Input: prime_check(5)
Output: TRUE
Input: prime_check(4)
Output: FALSE
Input: prime_check(237)
Output: FALSE
Input: prime_check(131)
Output: TRUE

```
prime_check <- function(num){
  if (num %% 2 == 0){
    return(TRUE)
  }
  return(FALSE)
}


prime_check(2)
prime_check(5)
prime_check(4)
prime_check(237)
prime_check(131)
```