PSTAT 10 Worksheet 3 Solutions

Problem 1: Contains Duplicate

Write the function `contains_duplicate(v)` that takes a numeric vector `v` and returns TRUE if any value appears **at least twice** in the vector and FALSE otherwise.

```
contains_duplicate <- function(v) {
  return(TRUE %in% duplicated(v))
  #duplicated(v) returns a boolean vector checking if each value of v has
  #duplicates, %in% return true if any of the values are found to be a duplicate
}
```

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

```
## [1] TRUE
```

```
contains_duplicate(c(1, 2, 3, 4))
```

```
## [1] FALSE
```

```
contains_duplicate(c(1, 1, 1, 3, 3, 4, 3, 2, 4, 2))
```

```
## [1] TRUE
```

Hint: One way is to use a loop and keep track of what elements you have seen. The `%in%` operator tests membership in a vector and could be helpful.

There is also an *extremely easy* way to do this using built-in R functionality.

Testing membership with `%in%`:

```
"cat" %in% c("dog", "cow", "cat", "owl")
```

```
## [1] TRUE
```

```
12 %in% c(3, 6, 1, 0)
```

```
## [1] FALSE
```

Problem 2: More on iris

For this section, we need the tidyverse library:

```
library(tidyverse)
```

1. Convert the `iris` data frame to a tibble and call it `iris_tbl`

```
data("iris")
iris_tbl <- as_tibble(iris)
```

2. Find the median `Petal.Width` and then create a tibble that only contains petal widths greater than the median.

```
median(iris_tbl$Petal.Width)
```

```
## [1] 1.3
```

The median petal width is 1.3.

```
iris_tbl |> filter(Petal.Width > median(Petal.Width)) |> select(Petal.Width)
```

```
## # A tibble: 72 x 1
##    Petal.Width
##          <dbl>
##  1         1.4
##  2         1.5
##  3         1.5
##  4         1.5
##  5         1.6
##  6         1.4
##  7         1.5
##  8         1.4
##  9         1.4
## 10         1.5
## # i 62 more rows
```

3. Call the area of a petal its length times its width. Create a tibble containing only the variables `Sepal.Length`, `Sepal.Width`, `Species`, and `Petal.Area` and only the rows where the petal width is greater than the median.

My result is the following:

```
# A tibble: 72 × 4
   Sepal.Length Sepal.Width Species    Petal.Area
          <dbl>       <dbl> <fct>           <dbl>
 1          7           3.2 versicolor       6.58
 2          6.4         3.2 versicolor       6.75
 3          6.9         3.1 versicolor       7.35
 4          6.5         2.8 versicolor       6.9
 5          6.3         3.3 versicolor       7.52
 6          5.2         2.7 versicolor       5.46
 7          5.9         3   versicolor       6.3
 8          6.1         2.9 versicolor       6.58
 9          6.7         3.1 versicolor       6.16
10          5.6         3   versicolor       6.75
# 62 more rows
# Use `print(n = ...)` to see more rows
```

```
iris_tbl |>
mutate(Petal.Area = Petal.Length*Petal.Width) |>
select(Sepal.Width, Sepal.Length, Species, Petal.Area) |>
filter(Petal.Area > median(Petal.Area))
```

```
## # A tibble: 75 x 4
##    Sepal.Width Sepal.Length Species    Petal.Area
##          <dbl>        <dbl> <fct>           <dbl>
##  1         3.2          7   versicolor       6.58
##  2         3.2          6.4 versicolor       6.75
##  3         3.1          6.9 versicolor       7.35
##  4         2.8          6.5 versicolor       6.9
##  5         2.8          5.7 versicolor       5.85
##  6         3.3          6.3 versicolor       7.52
##  7         2.9          6.6 versicolor       5.98
##  8         3            5.9 versicolor       6.3
```

```
##  9         2.9          6.1 versicolor       6.58
## 10         3.1          6.7 versicolor       6.16
## # i 65 more rows
```

Problem 3: More on heights data

Load the `heights_df` data frame from worksheet 1.

Recall the `height` variable is given in centimeters (cm). In worksheet 2, we created `cm_to_ft_inch` that converts from cm to a string representation of feet and inches.

Using `dplyr` functionality, create a tibble with a variable `height_ft_in` in place of `height`. The output is given:

```
# A tibble: 506 × 4
   `id_#` gender   age height_ft_in
    <dbl> <chr>  <dbl> <chr>
 1      1 Female    19 5 3
 2      2 Female    19 6 8
 3      3 Female    22 6 6
 4      4 Male      19 6 0
 5      5 Female    21 6 9
 6      6 Male      19 6 2
 7      7 Female    21 5 1
 8      8 Female    21 5 6
 9      9 Male      18 6 5
10     10 Female    18 5 5
# 496 more rows
# Use `print(n = ...)` to see more rows
```

```
cm_to_inch <- function(cm)
{
  inch <- cm * 0.3937
  return(inch)
}

cm_to_ft_inch <- function(cm)
{
  inch = cm_to_inch(cm)
  feet = inch %/% 12
  inch = inch %% 12
  return(paste(feet, round(inch), sep = " ", collapse = NULL))
}

heights_tbl <- as_tibble(read.csv("heights.csv"))
heights_tbl |>
mutate(height_ft_in=cm_to_ft_inch(height)) |>
select(id_., gender, age, height_ft_in)
```

```
## # A tibble: 506 x 4
##     id_. gender   age height_ft_in
##    <int> <chr>  <int> <chr>
##  1     1 Female    19 5 3
##  2     2 Female    19 5 8
##  3     3 Female    22 5 6
##  4     4 Male      19 6 0
```

```
##  5     5 Female    21 5 9
##  6     6 Male      19 6 2
##  7     7 Female    21 5 1
##  8     8 Female    21 5 6
##  9     9 Male      18 6 5
## 10    10 Female    18 5 5
## # i 496 more rows
```