sharks\_df['Age'] = sharks\_df['Age'].astype(str)  # Convert the column to string type

def clean\_age(age):

    if pd.isnull(age):

        return age  # Return NaN as it is if input is NaN

    age = age.replace("s", "")  # Remove 's'

    age = age.replace("!", "")  # Remove '!'

    age = age.replace("/", "")  # Remove '/'

    age = age.replace("&", "")  # Remove "&"

    age = age.replace(" or ", "")  # Replace "or"

    age = age.replace("and", "")  # Remove 'and'

    age = age.replace("?", "")

    age = age.replace("month", "1")

    age = age.replace("teen", "15")

    age = age.replace("Teen", "15")

    age = age.replace("Both", "")

    age = age.replace("young", "nan")

    age = age.replace("Elderly", "nan")

    age = age.replace("a minor", "nan")

    age = age.replace("nan", "nan")

    age = age.replace("(nan)", "nan")

    age = age.replace("adult", "nan")

    age = age.replace("(adult)", "nan")

    age = age.split(" or ")[0]  # Extract the first age if multiple ages are present

    return age

# Clean up the age entries

cleaned\_ages = [clean\_age(age) for age in sharks\_df['Age']]

# Assign the cleaned ages back to the 'Age' column

sharks\_df['Age'] = cleaned\_ages

# Calculate the average age excluding NaN values

average\_age = sharks\_df["Age"].dropna().astype(float).mean()

# Fill empty cells in the 'Age' column with the average age

sharks\_df["Age"].fillna(average\_age, inplace=True)

#Cleaning the "Age column"

# Convert to strings

sharks\_df['Age'] = sharks\_df['Age'].astype(str)

#Function to convert age ranges (20s, 30s, 40s, etc.) to middle value 20s = 25

def convert\_age\_range(age\_range):

  if age\_range.endswith('0s'):

    lower\_bound = int(age\_range[:-1]) + 5

# Replace age ranges (20s, 30s, 40s, etc.) with the middle value of the range

sharks\_df['Age'] = sharks\_df['Age'].str.replace(r'\(\d+s\)', lambda x: str(convert\_age\_range(x.group())))

# Funcion to remove special characters

def clean\_age(cell\_value):

  cell\_value = cell\_value.replace('!', '')

  if cell\_value.isnumeric():# Check if the cleaned cell is numeric

    return int(cell\_value) # Convert to integer if numeric

  else:

    return None # Return None for non-numeric values

# Function to check if the cell contains 'teen' and replace it with 15

def clean\_age(cell\_value):

  if 'teen' in str(cell\_value).lower():

    return 15

  else:     # Check if the cell contains any non-numeric characters and remove them

    cell\_value = ''.join(filter(str.isdigit, str(cell\_value)))

# Function to filter out cells containing multiple age entries or '&'

def process\_age(cell\_value): # Check if the cell value contains '/'

  if '/' in str(cell\_value):    # Check if the cell value contains more than one '/'

    if str(cell\_value).count('/') > 1:

      return None     # If there's only one '/', process the value

    else:

      parts = cell\_value.split('/')

      try:

        value\_sum = sum(map(int, parts))

        return value\_sum / 2

      except ValueError:

        return None # Return None if conversion to integer fails

  else:

    return cell\_value # Check if the cell contains '&', 'and', or 'or'

  if '&' in str(cell\_value) or 'and' in str(cell\_value) or 'or' in str(cell\_value):

    return None

  return cell\_value

# Apply the clean\_age function to the 'Age' column

sharks\_df['Age'] = sharks\_df['Age'].apply(clean\_age)

# Calculate the average age excluding NaN values

average\_age = sharks\_df["Age"].mean()

# Fill empty cells in the 'Age' column with the average age

sharks\_df["Age"].fillna(average\_age)