Remember Version 4 Reflection Activity
Q1 Modify the given code such that it eliminates non-adjacent duplicate line groups by implementing and then calling a user-defined function.

Given Code

```
def main():
    word_list = ['orange','apple','blueberry','kiwi']
    title = 'Print The Fruits'
    symbol = '*'
    message = 'The End'
    print(title)
    print(symbol)
    for word in word_list:
        print(title + ' ' + word)
    print(title)
    print(symbol)
    print(symbol + message + symbol)

main()
```

Modified Code with User-Defined Function

```
def display_header(title, symbol):
  print(title)
  print(symbol)

def main():
  word_list = [‘orange’, ‘apple’, ‘blueberry’, ‘kiwi’]
  title = ‘Print The Fruits’
  symbol = ‘*’
  message = ‘The End’
  display_header(title, symbol)
  for word in word_list:
    print(title + ‘ ‘ + word)
  display_header(title, symbol)
  print(symbol + message + symbol)

main()
```

Q2 Consider the program segment from Remember Version 4 to answer the following questions:

Given Program Segment

```
def prompt_for_guess(answer_start_letter):
    # prompts the user to guess the correct word
    display_header()
    guess=input('What word begins with the letter '+answer_start_letter+'? ')
    return guess
```

2.1 Underline all arguments and circle all parameters in the given program segment.

2.2 List each kind of statement in the given program segment.

functional expression statement, assignment statement, return statement

2.3 If a `return` statement is omitted from a function, then what type of object would be returned by that function?

NoneType

3. Complete this function, which takes a list of words as an argument and returns the longest in the list.

```
def longest_word(list_of_words):
   # returns the longest word in the list
   # - list_of_words is an object of type list
   longest_word = “”
   for word in list_of_words:
     if len(word) > len(longest_word):
       longest_word = word









    return longest_word
```

4. Rewrite the display_words function to take a second argument, which determines how long to sleep between displaying each word.

Original Code

```
def display_words(words):
    # Displays a list of words on screen. Each word is presented for 2 seconds.
    # The screen is cleared before and after each word presentation.
    for word in words:
        display_header()
        print(word)
        time.sleep(2)
```

Modified Code

```
def display_words(words, sleep_time):
  for word in words:
    display_header()
    print(word)
    time.sleep(sleep_time)
```

5. Consider how the function, sample_words(), works. Write a function that reads in a list of words from a file. It then returns a list of all words in that file that begin with a letter specified by the first parameter of the function. Note the sample code for creating a list and appending to it, below.

Sample Code

```
new_list = [ ] # this creates a new, empty list and binds it to new_list
new_list.append('foo') # this appends the str, 'foo', to the list
```

Original Code

```
def sample_words(n=4):
    # samples N words from our library and returns them as a list. All words
    # returned will start with a unique letter. Therefore, the maximum value of
    # n is 26.
    file = open("words.txt", "r")

    all_words = file.read()
    file.close()

    all_words = all_words.strip()
    all_words = all_words.splitlines()

    return random.sample(all_words, n)
```

Modified Code

```
def alphabetical_collection(start_letter):
    file = open("words.txt", "r")
```

```
def alpha_list(letter):
   filename = “word.txt”
   filemode = “r”
   file = file.open(filename, filemode)
   content = file.read()
   file.close()
   word_list = content.splitlines()
   letter_words = []
   for word in word_list:
      if word[0] == letter:
         letter_words.append(word)
   return letter_words

def main():
   start_letter = input(“Enter first letter >”)
   word_list = alpha_list(start_letter)
   print(word_list)

main()
```

6. For questions 3, 4, and 5, for each function you wrote, identify a) the type of the return value, and b) the type expected for each parameter.

Function Name	Expected argument types	Return type
Q~~4~~3, longest_word	list	str
Q~~5~~4, display_words	list, int	NoneType
Q~~6~~5, alphabetical collection	str	list