Section C Programming 50 marks

Answer all question parts.

Question 16

(a) Open the program called **Question16_A.py** from your device. The source code is shown below.

Before making any changes, you should save your working copy of the file using the format **CandidateNumberQuestion16_A.py**. For example, you would save the file as **123456Question16_A.py** if your candidate number was 123456.

Enter your Examination Number in the space provided on **line 2** in your Python file**.**

The Python program shown below tests whether the strings stored in the variables **`word1`** and **`word2`** are anagrams of each other and displays YES if they are.

Two words are anagrams of each other if they both use exactly the same letters. For example, the words LISTEN and SILENT are anagrams of one another. ELVIS and LIVES are also anagrams.

The function definition **`is_anagram(w1, w2)`** determines whether or not two words are anagrams of one another and will not be used until part **(v)**.

```
1  # Question 16(a)
2  # Examination Number:
3
4  # function definition used in part (v)
5  def is_anagram(w1, w2):
6      if sorted(w1) == sorted(w2):
7          return True
8      else:
9          return False
10
11 word1 = input("Enter the first word: ")
12 word2 = "SILENT"
13
14 # test whether the sorted strings are the same as each other
15 # if the sorted strings are the same then they must be anagrams
16 if (sorted(word1) == sorted(word2)):
17     print("YES")
```

When the program is run, the user is prompted to enter a word. If the user enters the word LISTEN (in uppercase) the output should look as follows because LISTEN and SILENT are anagrams of each other.

```
Enter the first word: LISTEN
YES
```

Make the following changes to the program:

(i) Currently the value of the variable **`word2`** is hard-coded to SILENT. Modify the program so that it prompts the user to ‘*Enter the second word:’,* and then assign whatever value the user enters to the variable **`word2`**.

When the program is run the output may look as follows:

```
Enter the first word: LISTEN
Enter the second word: SILENT
YES
```

(ii) If both words entered are anagrams the program should display the first word followed by the phrase ‘*is an anagram of*' and then the second word.

When the program is run the output may look as follows:

```
Enter the first word: LISTEN
Enter the second word: SILENT
LISTEN is an anagram of SILENT
```

(iii) Extend the program so that if the words entered are not anagrams of each other the program displays the first word followed by the phrase ‘*is NOT an anagram of*' and then the second word.

When the program is run the output may look as follows:

```
Enter the first word: LIST
Enter the second word: ARRAY
LIST is NOT an anagram of ARRAY
```

(iv) The program currently treats words as case-sensitive. Modify the program so that the case of the words entered does not matter.

When the program is run the output may look as follows:

```
Enter the first word: Listen
Enter the second word: Silent
Listen is an anagram of Silent
```

(v) Extend the program to use the function **`is_anagram`** to determine whether or not the two words entered are anagrams of each other. You should not delete any code you wrote in previous parts. The program will now check twice if the words are anagrams of each other.

When the program is run the output may look as follows:

```
Enter the first word: Listen
Enter the second word: Silent
Listen is an anagram of Silent
Listen is an anagram of Silent
```

(vi) Anagrams of words can also be phrases. For example, *Moon starer* is an anagram of *Astronomer,* and *Voices rant on* is an anagram of the two words, *Conversation* and *Conservation*. Note that the spaces in the phrases are ignored.

Extend the program so that it prompts the user to '*Enter a phrase:'*. The program should display two additional messages to say whether or not **`word1`** is an anagram of the phrase and, whether or not **`word2`** is an anagram of the phrase.

Some example outputs are shown below. You could use this data to test your program.

```
Enter the first word: Listen
Enter the second word: Silent
Listen is an anagram of Silent
Listen is an anagram of Silent

Enter a phrase: Silence of the lambs
Listen is NOT an anagram of Silence of the lambs
Silent is NOT an anagram of Silence of the lambs
```

```
Enter the first word: conversation
Enter the second word: conservation
conversation is an anagram of conservation
conversation is an anagram of conservation

Enter a phrase: voices rant on
conversation is an anagram of voices rant on
conservation is an anagram of voices rant on
```

```
Enter the first word: Astronomer
Enter the second word: Moon
Astronomer is NOT an anagram of Moon
Astronomer is NOT an anagram of Moon

Enter a phrase: moon starer
Astronomer is an anagram of moonstarer
Moon is NOT an anagram of moon starer
```

Save your file using the format **CandidateNumberQuestion16_A.py**. For example, you would save the file as **123456Question16_A.py** if your candidate number was 123456.

Space for rough work.

This page will not be reviewed by an examiner.