



Assignment 2:

“dits-dahs” Assignment

COURSE CODE : TTTK 1143
COURSE NAME : REKABENTUK ATUCARA
MATRIC NO/NAME : A175838 NG HAO LIN

1. Describes all data structures used in *dits-dahs* problem.

There are two data structures that I used in *dits-dahs* problem which are:

- i) Array
- ii) Binary Search Tree

For array, array is used to store the words after I split the space. After I add the words into the array, the words are used to search in binary search tree. I used array because an array is a data structure, which can store a fixed-size collection of elements of the same data type. An array is used to store a collection of data, but it is often more useful to think of an array as a collection of variables of the same type.

For binary search tree, it is used for the encode, decode and inorder part. For the encode part, the data type that I used is String. The character of alphabet, symbol and number are compared by using 'A.compareTo(B)'. The rule of insert word will follow the rules bellow:

- Data/key of left child < its parent, AND
- Data/key of right child > than its parent

Binary search tree is used for the encode part because it can improve searching time from linear structure. The searching way for binary search tree is fast. The searching way of BST:

- i. Start from root
- ii. Compare the inserting element(key) with root, if less than root, then recurse for left, else recurse for right.
- iii. If element to search is found anywhere, return true, else return false.

For the decode part, a Null root is created for the starting point. If the Morse Code starts form (.), it will move to the left side and if it is (-) and it will move to the right. From this part, when come to the searching part, the Morse Code also can search faster because the searching rule is same with the insert part which is:

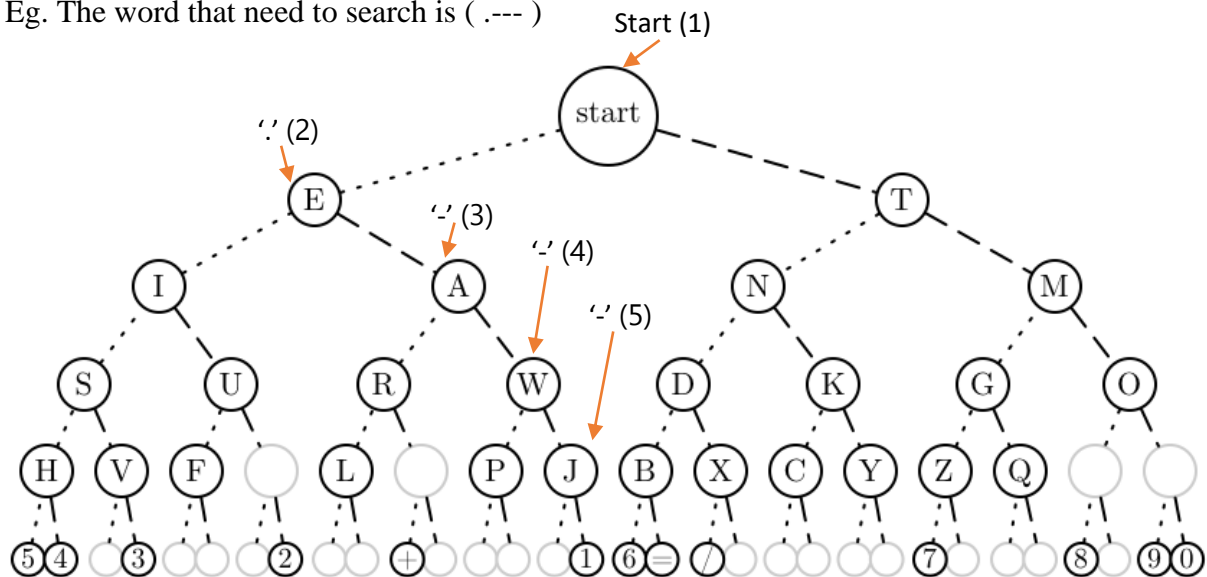
- If (.) move to the left
- If (-) move to the right

It can make the searching time faster. Thus, I use binary search tree in *dits-dahs* assignment.

2. Illustrates the structure of Morse Code after insert all the letters used.

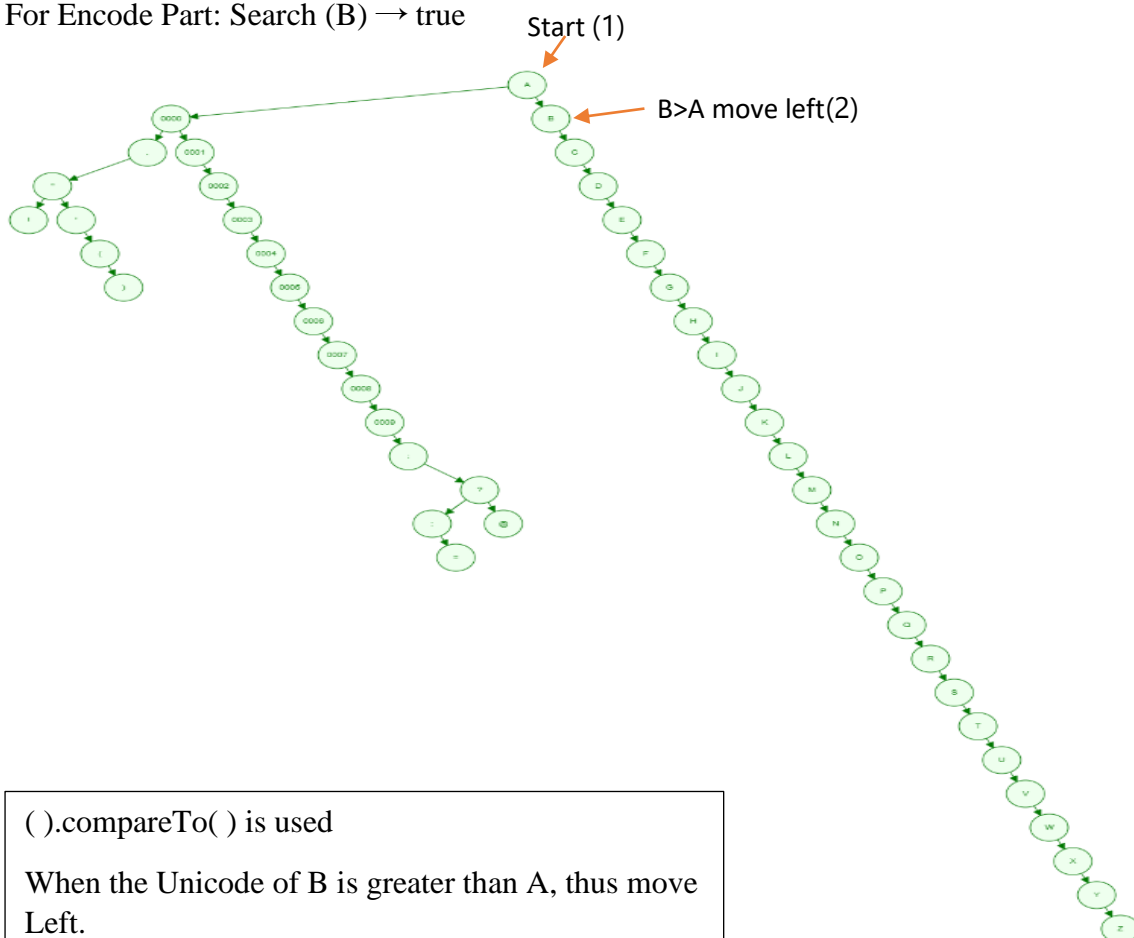
For decode part:

Eg. The word that need to search is (.---)



Search (.---) → true (it is J)

For Encode Part: Search (B) → true



().compareTo() is used

When the Unicode of B is greater than A, thus move Left.

3. Describes all classes used in *dits-dahs* (list all attributes and methods involve in each class and state the objective).

a) A175838_filename1 (Encode Part)

A175838_filename1	Description
key: String key1: String	- Variable to store letter - Variable to store Morse Code
+Class Node +A175838_filename() + insert(): void +insertRec(String, String): void Node insertRec1(Node, String, String) +search(String): String +inputSearch(String): String	- A BST is represented by a pointer to the topmost node in tree. If the tree is empty, then value of root is NULL -A constructor will initialize the members with the given value -Method mainly call super.insert() -Method mainly call insertRec1() -A recursive function to insert a new key in BST -Method mainly search the Morse Code -Method mainly call search -Method split the space of the word

b) A175838_filename2 (Decode Part)

A175838_filename2	Description
+A175838_filename2() +search(String): String +inputSearch(String): String	-A constructor will initialize the members with the given value -Method mainly search the letter of Morse Code -Method mainly call search -Method split the space of the word

c) A175838_filename3 (Inorder traversal && Filein Part)

A175838_filename3	Description
count: int key: String key1: String	-Variable to store the number of outputs for inorder traversal -Variable to store letter -Variable to store Morse Code

+Class Node	- A BST is represented by a pointer to the topmost node in tree. If the tree is empty, then value of root is NULL
+A175838_filename3()	- A constructor will initialize the members with the given value
+insert(): void	-Method mainly insert data from .dat file -Method mainly call insertRec()
+insertRec(String, String): void	-Method mainly insert a new key in BST
+inorder(): void	-Method mainly call inorderRec()
+inorderRec(Node): void	-Method mainly print inorder traversal of BST
+counter(int): void	-Method mainly count the number of output and create a new line for the outputs of inorder traversal

d) A175838_Mainfile (Main File)

A175838_Mainfile	Description
+main: void	-Main class to call all the class
+menu: void	-Method to print out the menu
+wordcount(String): int	-Method to count the number of words
+charactercount(String):int	-Method to count the character of words
+symbolcount(String): int	-Method to count the number of symbols
+numbercount(String): int	-Method to count the value of number

4. Algorithm for:

i. Main method

1. If click 1, reach encode part
2. If click 2, reach decode part
3. If click 3, reach inorder traversal part
4. If click 4, program end

ii. Encode method

1. Visit the root
2. If greatest than root, then recurse for left, else recurse for right
3. If element is found, return true, else return false

iii. Decode method

1. Start from the null root
2. If dot(.), then recurse for right
3. If dash(-), then recurse for left
4. If element is found, return true, else return false

iv. Display all letters and Morse code

Inorder traversal (LNR):

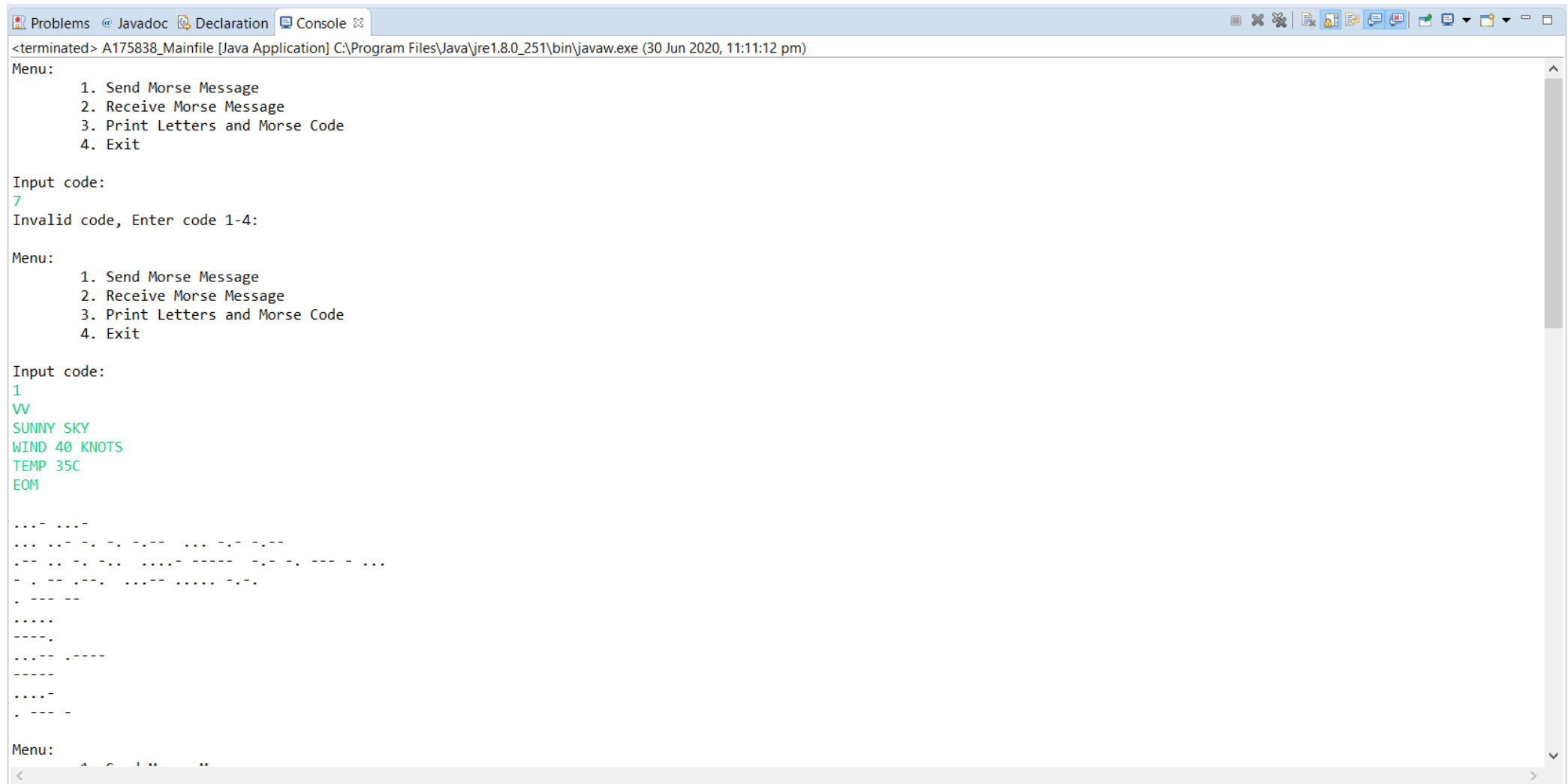
1. Traverse the left subtree
2. Visit the root.
3. Traverse the right subtree

5. Any assumptions that I made.

- i) If type the number without 1 until 4 in the menu place, the program will ask to type the number again.
- ii) For the encode part (1), the program will require to start with VV.
- iii) The output of encode (Morse Code) will follow the rule: Letters are separated by a space, while one word to another are separated by two spaces.
- iv) The program will stop to encode the letter when receive EOM.
- v) For the decode part (2), the program will require to start with ...- ...-
- vi) The input of decode (Morse Code) need to follow the rule: Letters are separated by a space, while one word to another are separated by two spaces.
- vii) The program will stop to decode the Morse Code when receive . --- -
- viii) For the decode part, a summary analysis generated based on the decoded message to be compared with the transmission summary from the sender. The comparison will print out "Consistent Summary" if the two messages equal, otherwise print "Inconsistent Summary".
- ix) For the inorder traversal part (3), the program will print five output in a row.

6. Screenshot Input and Output

A)



```
<terminated> A175838_Mainfile [Java Application] C:\Program Files\Java\jre1.8.0_251\bin\javaw.exe (30 Jun 2020, 11:11:12 pm)
Menu:
    1. Send Morse Message
    2. Receive Morse Message
    3. Print Letters and Morse Code
    4. Exit

Input code:
7
Invalid code, Enter code 1-4:

Menu:
    1. Send Morse Message
    2. Receive Morse Message
    3. Print Letters and Morse Code
    4. Exit

Input code:
1
VV
SUNNY SKY
WIND 40 KNOTS
TEMP 35C
EOM

...- ...-
... ..- .- .- .- .- .- .-
- .- .- .- .- .- .- .- .-
- .- .- .- .- .- .- .-
. - - - -
....
- - - .
...- - - -
- - - -
...-
. - - - -

Menu:
```

B)

The screenshot shows a Java application window titled "A175838_Mainfile [Java Application]". The window has a menu bar with "Problems", "Javadoc", "Declaration", and "Console". The console output shows the following:

```
<terminated> A175838_Mainfile [Java Application] C:\Program Files\Java\jre1.8.0_251\bin\javaw.exe (30 Jun 2020, 11:11:12 pm)
Menu:
1. Send Morse Message
2. Receive Morse Message
3. Print Letters and Morse Code
4. Exit

Input code:
2
...- ...-
... ..- .- -.- -.- -.-
..- ..- ..- ..- ..- ..- ..-
.- ..- ..- ..- ..-
.- ..- ..-
....
-----
.....- .....
-----
.....-
.- ..-

VV
SUNNY SKY
WIND 40 KNOTS
TEMP 35C
EOM
5
9
31
0
4
EOT

5 9 31 0 4
Result: Consistent Summary

Menu:
1. Send Morse Message
```


C)

```
Problems @ Javadoc Declaration Console
<terminated> A175838_Mainfile [Java Application] C:\Program Files\Java\jre1.8.0_251\bin\javaw.exe (30 Jun 2020, 11:11:12 pm)
31
0
4
EOT

5 9 31 0 4
Result: Consistent Summary

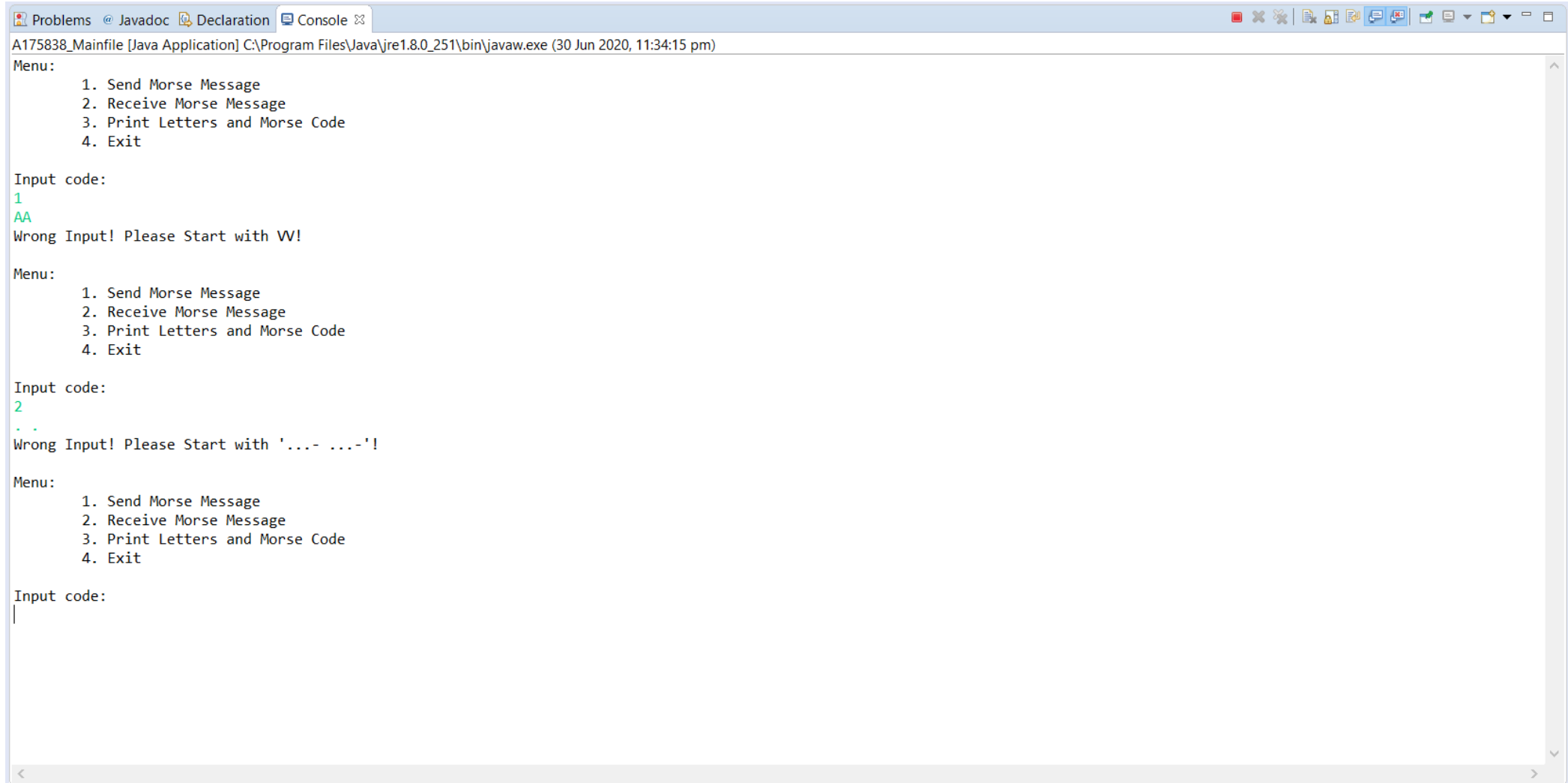
Menu:
  1. Send Morse Message
  2. Receive Morse Message
  3. Print Letters and Morse Code
  4. Exit

Input code:
3
5 .....      H ....      4 ....-      S ...      V ...-
3 ....-      I ..      F ...      U ..-      ? .....
2 ..-       E .      L ...      " .....      R ...
. ....-      A .-      P ...      @ .....      W ...-
J ....      ' .....      1 .....      6 .....      - .....-
B ....      = .....-      D ...      X ....-      N ...
C ...      ; .....      ! .....      K ...      ( .....
) .....-      Y ...      T ...      7 .....      Z ...
, .....-      G ...      Q ...-      M ...      : .....
8 .....      O ...      9 .....      0 .....

Menu:
  1. Send Morse Message
  2. Receive Morse Message
  3. Print Letters and Morse Code
  4. Exit

Input code:
4
Bye dits-dahs..
```

D)



```
Problems @ Javadoc Declaration Console
A175838_Mainfile [Java Application] C:\Program Files\Java\jre1.8.0_251\bin\javaw.exe (30 Jun 2020, 11:34:15 pm)

Menu:
  1. Send Morse Message
  2. Receive Morse Message
  3. Print Letters and Morse Code
  4. Exit

Input code:
1
AA
Wrong Input! Please Start with W!

Menu:
  1. Send Morse Message
  2. Receive Morse Message
  3. Print Letters and Morse Code
  4. Exit

Input code:
2
. .
Wrong Input! Please Start with '...- ...-'!

Menu:
  1. Send Morse Message
  2. Receive Morse Message
  3. Print Letters and Morse Code
  4. Exit

Input code:
|
```

E)

```
Input code:  
.  
. .  
. . .  
. . . .  
. . . . .  
. . . . .  
  
.  
.  
.  
.  
.  
.  
.  
.  
  
VV  
SUNNY SKY  
WIND 40 KNOTSE  
TEMP 35C  
EOM  
5  
9  
31  
0  
4  
EOT  
  
5 9 32 0 4  
Result: Inconsistent Summary  
  
Menu:  
    1. Send Morse Message  
    2. Receive Morse Message  
    3. Print Letters and Morse Code  
    4. Exit  
  
Input code:
```