/\*

\* Justin Mendes

\* Created: September 14, 2017

\* Last Edited: September 14, 2017

\* Unit 1 Activity 2 Program/Question 1

\* This program will show the unicode values of letters a-z uppercase and lowercase

\*/

**public** **class** UnicodeChars

{

**public** **static** **void** main(String[] args)

{

//Variable Declarations and Initializations

**char** alphabet[][] = {

{'A', 'a'}, {'B', 'b'}, {'C', 'c'}, {'D', 'd'}, {'E', 'e'}, {'F', 'f'}, {'G', 'g'}, {'H', 'h'}, {'I', 'i'}, {'J', 'j'}, {'K', 'k'},

{'L', 'l'}, {'M', 'm'}, {'N', 'n'}, {'O', 'o'}, {'P', 'p'}, {'Q', 'q'}, {'R', 'r'}, {'S', 's'}, {'T', 't'}, {'U', 'u'}, {'V', 'v'},

{'W', 'w'}, {'X', 'x'}, {'Y', 'y'}, {'Z', 'z'}};

System.***out***.println("Unicodes of the Alphabet\n=============================");

**for**(**int** letter = 0; letter < 26; letter++)

{

**for**(**int** cases = 0; cases < 2 ; cases++)

{

System.***out***.print(alphabet[letter][cases] + " = \\u" + Integer.*toHexString*(alphabet[letter][cases] | 0x10000).substring(1) + ", ");

//The toHextString(int i) method simply returns the hexadecimal (or base 16) string equivalent of int method parameter.

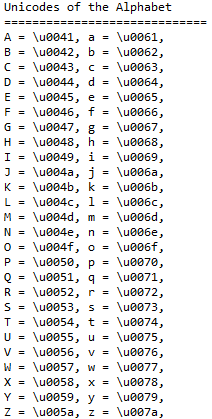
}//end loop

System.***out***.println();

}//end loop

}//end main

}//end class



/\*

\* Justin Mendes

\* Created: November 26, 2016

\* Last Edited: September 17, 2017

\* Unit 1 Activity 2 Program/Question 3

\* This program will print the user's inputed word backward and declare if the word is a palindrome (spelt the same backwards as forward)

\*/

**import** javax.swing.JOptionPane;

**import** java.util.Scanner;

**public** **class** Palindrome

{

**public** **static** **void** main(String[] args)

{

**int** restart = 1;

**while**(restart == 1)

{

//Variable Declarations and Initializations

String backward = "", palindrome;

Scanner sc = **new** Scanner(System.***in***);

palindrome = JOptionPane.*showInputDialog*(**null**,"Words that are the same forwards and backwards are called palindromes.\nThis program determines if a word is a palindrome.\n\nEnter a word.", "Input", JOptionPane.***QUESTION\_MESSAGE***);

System.***out***.print(palindrome.toLowerCase() + " backwards is ");

**for** (**int** i = palindrome.length() - 1; i >= 0; i--)

{

backward += palindrome.charAt(i);

}//end for

System.***out***.print(backward);

**if** (*palindromeCheck*(palindrome) == **true**)

{

System.***out***.println("\nTherefore, " + palindrome + " IS a palindrome!");

}//end if

**else**

{

System.***out***.println("\nClearly, " + palindrome + " is NOT a palindrome.");

}

System.***out***.println("\nPress 1 to try another word!");

restart = sc.nextInt();

}//end while

}//end main

**public** **static** **boolean** palindromeCheck(String word)

{

String backward = "";

**for** (**int** i = word.length() - 1; i >= 0; i--)//loop to have the word backwards

{

backward += word.charAt(i);

}//end for

**if**(backward.toLowerCase().equals(word.toLowerCase()))//the word is palindrome if it is the same backwards

{

**return** **true**;

}//end if

**else**

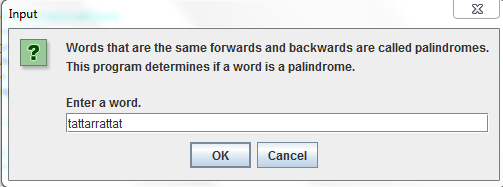
{

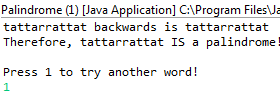
**return** **false**;

}//end else

}//end method palindromeCheck

}//end class





/\*

\* Justin Mendes

\* Created: September 17, 2017

\* Last Edited: September 17, 2017

\* Unit 1 Activity 2 Program/Question 4

\* This program can print the user's inputed sentence words backward and declare if the word is a palindrome (spelt the same backwards as forward)

\*/

**import** javax.swing.JOptionPane;

**import** java.util.Scanner;

**public** **class** Palindrome2

{

**public** **static** **int** *palindromeCount*;

**public** **static** **void** main(String[] args)

{

**int** restart = 1;

**while**(restart == 1)

{

//Variable Declarations and Initializations

String sentence, palindromeWords[], palindromes = "";

Scanner sc = **new** Scanner(System.***in***);

sentence = JOptionPane.*showInputDialog*(**null**,"Words that are the same forwards and backwards are called palindromes.\nThis program determines if a word is a palindrome.\n\nEnter a sentence(do not include a punctuation mark):", "Input", JOptionPane.***QUESTION\_MESSAGE***);

palindromeWords = sentence.split(" ");// to split up the sentence by every space into the array

*palindromeCount* = 0;//to reset the palindrome count

//loop a palindrome check for each word

**for** (**int** i = 0; i < palindromeWords.length; i++)

{

palindromes += *palindromeCheck*(palindromeWords[i]) + " ";

}//end loop

System.***out***.println("Palindrome 2: Sentences\n===========================");

System.***out***.println("In the following sentence:\n" + sentence);

System.***out***.println("There are " + *palindromeCount* + " palindromes which are:\n" + palindromes.replaceAll("\\s+"," ")/\*this is to get rid of the excess spaces made from the code above\*/);

System.***out***.println("\nPress 1 to try another sentence!");

restart = sc.nextInt();

}//end while

}//end main

**public** **static** String palindromeCheck(String word)

{

String backward = "";

**for** (**int** i = word.length() - 1; i >= 0; i--)//loop to have the word backwards

{

backward += word.charAt(i);

}//end for

**if**(backward.toLowerCase().equals(word.toLowerCase()))//the word is palindrome if it is the same backwards

{

*palindromeCount*++;

**return** word;

}//end if

**else**

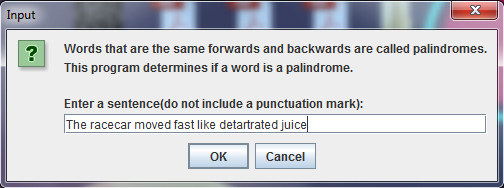
{

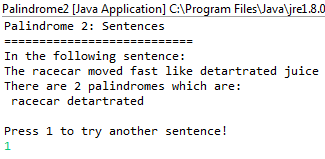
**return** "";

}//end else

}//end method palindromeCheck

}//end class





/\*

\* Justin Mendes

\* Created: September 17, 2017

\* Last Edited: September 17, 2017

\* Unit 1 Activity 2 Program/Question 5

\* This program will print the user's inputed phrase backwards and declare if the phrase is a palindrome (spelt the same backwards as forward)

\*/

**import** javax.swing.JOptionPane;

**import** java.util.Scanner;

**public** **class** Palindrome3

{

**public** **static** **void** main(String[] args)

{

**int** restart = 1;

**while**(restart == 1)

{

//Variable Declarations and Initializations

String sentence;

Scanner sc = **new** Scanner(System.***in***);

sentence = JOptionPane.*showInputDialog*(**null**,"Words that are the same forwards and backwards are called palindromes.\nThis program determines if a phrase is a palindrome.\n\nEnter a phrase(do not include a punctuation mark):", "Input", JOptionPane.***QUESTION\_MESSAGE***);

System.***out***.println("Palindrome 3: Whole Phrases\n================================");

**if**(*palindromeCheck*(sentence.replaceAll("\\s+", "")) == **true**)//give the user input without the spaces to check if the letters together make a palindrome

{

System.***out***.println(sentence + " IS a palindrome!");

}//end if

**else**

{

System.***out***.println(sentence + " is NOT a palindrome.");

}//end else

System.***out***.println("\nPress 1 to try another phrase!");

restart = sc.nextInt();

}//end while

}//end main

**public** **static** **boolean** palindromeCheck(String word)

{

String backward = "";

**for** (**int** i = word.length() - 1; i >= 0; i--)//loop to have the word backwards

{

backward += word.charAt(i);

}//end for

**if**(backward.toLowerCase().equals(word.toLowerCase()))//the word is palindrome if it is the same backwards

{

**return** **true**;

}//end if

**else**

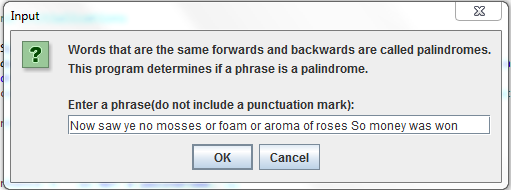
{

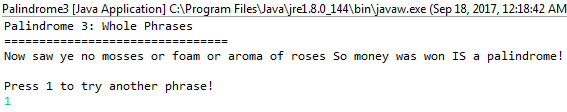
**return** **false**;

}//end else

}//end method palindromeCheck

}//end class





**import** javax.swing.JOptionPane;

/\*

\* Justin Mendes

\* Created: July 30, 2017

\* Last Edited: September 16, 2017

\* Unit 1 Activity 1 Program/Question 6

\* This program will encrypt a word or phrase based on the inputs

\*/

**public** **class** SimpleEncryption

{

**public** **static** **void** main(String[] args)

{

//Variable Declarations and Initializations

String userCode;

**int** method = 0, rotation = 0;

userCode = JOptionPane.*showInputDialog*(**null**,"Morse Code\n=============\n\nEnter a word or phrase to be encrypted or decrypted"

+ "\nusing the simple encryption method of rotating the letters", "Simple Encryption", JOptionPane.***QUESTION\_MESSAGE***);

**while** (rotation < 1 || rotation > 25)

{

rotation = Integer.*parseInt*(JOptionPane.*showInputDialog*(**null**,"Enter the rotation amount (1-25)", "Simple Encryption", JOptionPane.***QUESTION\_MESSAGE***));

}//end loop

**while** (method < 1 || method > 2)

{

method = Integer.*parseInt*(JOptionPane.*showInputDialog*(**null**,"1 - Encryption\n2 - Decryption", "Simple Encryption", JOptionPane.***QUESTION\_MESSAGE***));

}//end loop

System.***out***.println("Encryption\n==========");

System.***out***.println("The original phrase is: " + userCode);

System.***out***.println("The encrypted phrase is: " + *letterSwitch*(userCode, rotation, method));

}//end main

**public** **static** String letterSwitch(String userCode, **int** rotations, **int** method)

{

**char** alphabet[] = {'A','B','C','D','E','F','G','H','I','J','K',

'L','M','N','O','P','Q','R','S','T','U','V','W','X','Y','Z'};

String alteredCode = "";

//to go through each character in the string

**for**(**int** letterIdx = 0; letterIdx < userCode.length(); letterIdx++)

{

//to match up where the letter is in the array

**for**(**int** i = 0; i < 26; i++)

{

**if**(userCode.charAt(letterIdx) == ' ')//to ensure spaces work in phrases

{

alteredCode += " ";

**break**;

}//end if

**if**(alphabet[i] == userCode.toUpperCase().charAt(letterIdx))

{

**if**(method == 1)

{

alteredCode += *encrypt*(alphabet, i, rotations);

}//end if

**else**

{

alteredCode += *decrypt*(alphabet, i, rotations);

}//end else

}//end if

}//end loop

}//end loop

**return** alteredCode;

}//end method encryptDecrypt

**public** **static** **char** encrypt(**char**[] alphabet, **int** currentLetter, **int** rotations)

{

**if**(currentLetter + rotations >= 26)

{

**return** alphabet[currentLetter + rotations - 26];

}//end if

**else**

{

**return** alphabet[currentLetter + rotations];

}//end else

}//end method encrypt

**public** **static** **char** decrypt(**char**[] alphabet, **int** currentLetter, **int** rotations)

{

**if**(currentLetter - rotations < 0)

{

**return** alphabet[currentLetter - rotations + 26];

}//end if

**else**

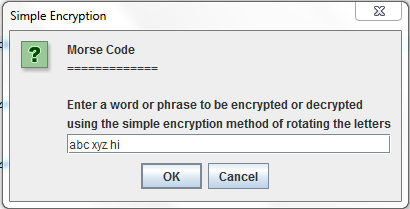
{

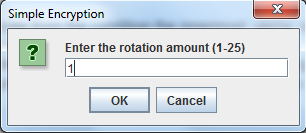
**return** alphabet[currentLetter - rotations];

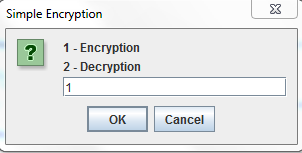
}//end else

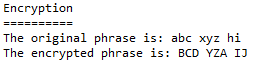
}//end method decrypt

}//end class









**import** javax.swing.JOptionPane;

/\*

\* Justin Mendes

\* Created: July 30, 2017

\* Last Edited: September 17, 2017

\* Unit 1 Activity 1 Program/Question 7

\* This program will return an inputted word in morse code

\*/

**public** **class** MorseCode {

**public** **static** **void** main(String[] args)

{

String input, code = "", upperInput;

input = JOptionPane.*showInputDialog*(**null**,"Morse Code\n=============\n\nEnter a word or phrase to make into morse code!"

+ "\n(spaces will be represented by /'s)", "Morse Converter", JOptionPane.***QUESTION\_MESSAGE***);

upperInput = input.toUpperCase();

**for**(**int** i = 0; i < input.length(); i++)

{

**switch**(upperInput.charAt(i))

{

**case** 'A': code += ".- ";

**break**;

**case** 'B': code += "-... ";

**break**;

**case** 'C': code += "-.-. ";

**break**;

**case** 'D': code += "-.. ";

**break**;

**case** 'E': code += ". ";

**break**;

**case** 'F': code += "..-. ";

**break**;

**case** 'G': code += "--. ";

**break**;

**case** 'H': code += ".... ";

**break**;

**case** 'I': code += ".. ";

**break**;

**case** 'J': code += ".--- ";

**break**;

**case** 'K': code += "-.- ";

**break**;

**case** 'L': code += ".-.. ";

**break**;

**case** 'M': code += "-- ";

**break**;

**case** 'N': code += "-. ";

**break**;

**case** 'O': code += "--- ";

**break**;

**case** 'P': code += ".--. ";

**break**;

**case** 'Q': code += "--.- ";

**break**;

**case** 'R': code += ".-. ";

**break**;

**case** 'S': code += "... ";

**break**;

**case** 'T': code += "- ";

**break**;

**case** 'U': code += "..- ";

**break**;

**case** 'V': code += "...- ";

**break**;

**case** 'W': code += ".-- ";

**break**;

**case** 'X': code += "-..- ";

**break**;

**case** 'Y': code += "-.-- ";

**break**;

**case** 'Z': code += "--.. ";

**break**;

**case** '1': code += ".---- ";

**break**;

**case** '2': code += "..--- ";

**break**;

**case** '3': code += "...-- ";

**break**;

**case** '4': code += "....- ";

**break**;

**case** '5': code += "..... ";

**break**;

**case** '6': code += "-.... ";

**break**;

**case** '7': code += "--... ";

**break**;

**case** '8': code += "---.. ";

**break**;

**case** '9': code += "----. ";

**break**;

**case** '0': code += "----- ";

**break**;

**case** ' ': code += " / ";

**break**;

}//end switch

}//end loop

System.***out***.println("Morse Code\n===============");

System.***out***.println("The word/phrase \"" + input + "\" in morse code is:");

System.***out***.println(code);

}//end main

}//end class

