



COMP 1039

Problem Solving and Programming

Programming Assignment 1

Prepared by
Jo Zucco

SAIBT Modifications
Andreas Jordan and Jane Emmett

School of Computer and Information Science
The University of South Australia
June 2017

Contents

| | |
|--|-----------|
| INTRODUCTION | 1 |
| ASSIGNMENT OVERVIEW | 1 |
| Bonus Marks | 2 |
| A Word on Functions... | 2 |
| PART I SPECIFICATION – PETALS AROUND THE ROSE | 3 |
| PRACTICAL REQUIREMENTS (PART I) | 6 |
| STAGES (PART I) | 7 |
| Stage 1 | 7 |
| Stage 2 | 7 |
| Stage 3 | 8 |
| Stage 4 | 8 |
| Stage 5 | 9 |
| Stage 6 | 10 |
| Stage 7 | 10 |
| Stage 8 | 10 |
| Stage 9 | 11 |
| Stage 10 | 12 |
| PART II SPECIFICATION – NUMBER CONVERTER | 13 |
| PRACTICAL REQUIREMENTS (PART II) | 14 |
| STAGES (PART II) | 15 |
| Stage 1 | 15 |
| Stage 2 | 16 |
| Stage 3 | 17 |
| Stage 4 | 17 |
| Stage 5 | 17 |
| Stage 6 | 17 |
| SUBMISSION DETAILS | 18 |
| EXTENSIONS AND LATE SUBMISSIONS | 19 |
| ACADEMIC MISCONDUCT | 19 |
| MARKING CRITERIA | 20 |
| SAMPLE OUTPUT – PART I | 22 |
| SAMPLE OUTPUT – PART II | 34 |

INTRODUCTION

This document describes the first assignment for Problem Solving and Programming.

The assignment is intended to provide you with the opportunity to put into practice what you have learnt in the course by applying your knowledge and skills to the implementation of a game/puzzle called **Petals Around the Rose** and **a simple number converter**.

This assignment is an **individual task** that will require an **individual submission**. **You will be required to present your work to your practical supervisor during your practical session held in Week 8 of the study period.** Important: You must attend the practical session that you have been attending all study period in order to have your assignment marked.

This document is a kind of specification of the required end product that will be generated by implementing the assignment. Like many specifications, it is written in English and hence will contain some imperfectly specified parts. Please make sure you seek clarification if you are not clear on any aspect of this assignment.

ASSIGNMENT OVERVIEW

There are two parts to this assignment:

Part I: Petals Around the Rose

You are required to write a Python program that allows a user to play a game called Petals Around the Rose. The program allows the user to repeatedly guess the answer to the puzzle until the user chooses to stop guessing/playing. Once the user chooses to stop guessing, the program will report the user's and game play statistics to the screen. You may like to play a web version of the game: <http://www.borrett.id.au/computing/petals-j.htm>.

Part II: Number converter

You are required to write a Python program that performs decimal to binary conversion (and vice versa) on numbers entered by the user. You are only required to deal with whole numbers. If you are interested, you may like to read the following on binary numbers: http://en.wikipedia.org/wiki/Binary_number.

"There are only 10 types of people in the world: those who understand binary, and those who don't."

– http://en.wikipedia.org/wiki/Mathematical_joke

Please ensure that you read sections titled 'Part I Specification' and 'Part II Specification' below for further details.

Bonus Marks

Since we do not begin to cover python functions until Week 7, the assignment does not require you to implement functions. However, an opportunity exists for those wanting an additional challenge in order to gain bonus marks. Note that you can achieve no higher than 100% for your assignment.

Bonus marks will be awarded as follows:

- Part I – 2 marks
- Part II – 2 marks

In order to be awarded any bonus marks, you will need to determine which parts of your code can be decomposed and placed into functions. Note that simply placing some bits of code into a function without a clear purpose will not receive bonus marks. You must use appropriate names for functions and each function must serve a specific purpose.

A Word on Functions...

In order to solve problems, it may be easier to separate code into smaller, simpler tasks. This process is called decomposition. A function is a self-contained segment of code which implements a specific well-defined task. Python programs typically combine user-defined functions with library functions available in the standard library.

A function is invoked by a call which specifies:

- A function name,
- May provide parameters/arguments.
- A function may return a value.

The general syntax for defining a function is:

```
def functionName(parameters):  
    function_body_suite
```

Example: A function that takes two numbers as parameters, sums them, and returns the result

```
def sum(num1, num2):  
    result = num1 + num2  
    return result
```

If you are interested in implementing functions, I encourage you to read the lecture slides for Weeks 7 and 8.

PART I SPECIFICATION – PETALS AROUND THE ROSE

You are required to write a Python program called *yourSaibtId.pets.py* that allows a user to play a game called **Petals Around the Rose**.

Petals Around the Rose

You are required to write a Python program that allows a player to play a game of Petals Around the Rose. The program allows the user to repeatedly guess the answer to the puzzle until the user chooses to stop guessing/playing. Once the user chooses to stop guessing, the program will report the game statistics to the screen. You may also like to read Bill Gates and Petals Around the Rose: <http://www.borrett.id.au/computing/petals-bg.htm>.

We will be adhering to the following ‘Petals Around the Rose’ rules and game play for the assignment.

Petals Around the Rose Game Play and Rules:

The name of the game is Petals Around the Rose and the name of the game is important. The computer will roll five dice and ask the user to guess the score for the roll. The score will always be an even number (including zero). The user’s mission is to work out how the computer calculates the score in order to become a Potentate of the Rose.

- To begin, the following instructions are displayed to the screen and the user is asked whether they would like to play Petals Around the Rose, i.e.:

```
Petals Around the Rose
-----
The name of the game is 'Petals Around the Rose'. The name of the
game is important. The computer will roll five dice and ask you to
guess the score for the roll. The score will always be zero or an
even number. Your mission, should you choose to accept it, is to
work out how the computer calculates the score. If you succeed in
working out the secret and guess correctly four times in a row, you
become a Potentate of the Rose.

Would you like to play Petals Around the Rose [y|n]?
```

If the user enters ‘n’, the following message is displayed to the screen only:

```
No worries... another time perhaps... :)
```

If the user enters ‘y’, game play continues as normal.

- Game play is as follows:

- The dice face values are then displayed to the screen and the user is asked to guess the score for the roll, for example:

```

Die 1      Die 2      Die 3      Die 4      Die 5
[4]        [3]        [2]        [5]        [6]

* *        *          *          * *        * *
          *          *          *          * *
* *        *          *          * *        * *

```

Please enter your guess for the roll:

The numbers in the above output will change depending on the dice values rolled.

- If the user guesses correctly, the following message is displayed:

Well done! You guessed it!

- If the user does not guess correctly one of two possible messages is displayed to the screen. If the user enters an incorrect non-even number, the following message is displayed to the screen, for example:

No sorry, it's 0 not 9. The score is always even.

- If the user enters an incorrect even number, the following message is displayed to the screen, for example:

No sorry, it's 0 not 6.

The numbers in the above output will change depending on what the user has entered and the score value for the roll.

- The user is then asked whether they would like to play again with the following prompt. Game play continues while the user enters 'y' at the prompt:

Roll dice again [y|n]?

- If the user has four (4) **incorrect** guesses **in a row** the following message is displayed to the screen (once only and then the count resets back to zero):

Hint: The name of the game is important... Petals Around the Rose

- If the user has four (4) **correct** guesses **in a row**, the following message is displayed to the screen (once only and then the count resets back to zero):

Congratulations! You have worked out the secret!
Make sure you don't tell anyone!

- Game play continues until the user chooses to quit, i.e. enters 'n' at the following prompt:

Roll dice again [y|n]?

- The solution or formula for Petals Around the Rose can be found here:

https://en.wikipedia.org/wiki/Petals_Around_the_Rose

The key to working out the answer is given by the name of the puzzle, where the “rose” is the centre dot that appears only on the 3 and 5 faces of a die, while the “petals” are the dots which surround the centre dot. Therefore, the result can be calculated by counting 2 for each 3 face and 4 for each 5 face. In the example provided above, there is one 5 face and one 3 face, so the result is four plus two, resulting in a total of six.

- Once the user chooses to quit (after having played at least one game), the game summary is displayed to the screen, for example:

```
Game Summary
=====
```

```
You played 2 games:
```

```
|--> Number of correct guesses:  1
```

```
|--> Number of incorrect guesses: 1
```

```
Dice Roll Stats:
```

```
Face  Frequency
```

```
1    *
```

```
2    ***
```

```
3    *
```

```
4    *
```

```
5    **
```

```
6    **
```

```
Thanks for playing!
```

The numbers in the above output will change depending on the results of the user's games. The above should not be displayed if the user has not played any games.

You do not have to write the code that displays the die face values to the screen, a module containing a function that does that for you has been provided. The `dice.py` file is a module that contains a function called `display_dice` that displays the face values of the dice to the screen for you. You are required to use this as part of this assignment, however, **please do not modify the `dice.py` file.**

PRACTICAL REQUIREMENTS (PART I)

It is recommended that you develop this part of the assignment in the suggested stages.

It is expected that your solution WILL include the use of:

- Your solution in a file called `yourSaibtId_petals.py`.
- The supplied `dice.py` module (containing the `display_dice` function). This is provided for you – **please DO NOT modify this file**.
- Appropriate and well constructed `while` and/or `for` loops (as necessary).
- Appropriate `if`, `if-else`, `if-elif-else` statements (as necessary).
- The use of the `random.randint(1,6)` function in order to simulate the roll of a six sided die.
- One function (refer to stage 1 for description) called `display_dice()`.
- Output that **strictly** adheres to the assignment specifications. If you are not sure about these details, you should check with the ‘Sample Output – Part I’ provided at the end of this document.
- Good programming practice:
 - Consistent commenting, layout and indentation. You are to provide comments to describe: your details, program description, all variable definitions, and every significant section of code.
 - Meaningful variable names.

Your solutions **MUST NOT** use:

- `break`, or `continue` statements in your solution. Do not use the `quit()` or `exit()` functions or the `break` or `return` statements (or any other techniques) as a way to break out of loops. Doing so will result in a significant mark deduction.

PLEASE NOTE: You are reminded that you should ensure that all input and output conform to the specifications listed here; if you are not sure about these details you should check with the sample output provided at the end of this document or post a message to the discussion forum in order to seek clarification.

Please ensure that you use Python 3.5.2 or a later version (i.e. the latest version) in order to complete your assignments. Your programs **MUST** run using Python 3.5.2 (or latest version).

STAGES (PART I)

It is recommended that you develop this part of the assignment in the suggested stages. Many problems in later stages are due to errors in early stages. **Make sure you have finished and thoroughly tested each stage before continuing.**

The following stages of development are recommended:

Stage 1

You will need the `dice.py` file for this assignment. This has been provided for you. Please download this file from the course website (Assignment tab) and ensure that it is in the same directory as the `yourSaibtId_petals.py` file.

Test to ensure that this is working correctly by entering the following in your `yourSaibtId_petals.py` file:

```
import dice

dice.display_dice(2, 3, 4, 5, 6)
```

Run the `yourSaibtId_petals.py` file. If this is working correctly, you should now see the following output in the Python shell when you run your program:

```
Die 1      Die 2      Die 3      Die 4      Die 5
[2]        [3]        [4]        [5]        [6]

*          *          * *        * *        * *
          *          *          *          * *
*          *          * *        * *        * *
```

Note, this is for developmental purposes only, and you will need to modify and correctly position the above code.

Make sure the program runs correctly. Once you have that working, back up your program. *Note: When developing software, you should always have fixed points in your development where you know your software is bug free and runs correctly.*

Stage 2

Add code to simulate the rolling of five dice. Use the `random.randint(1,6)` function to simulate the roll of a die.

Modify your code to now display the randomly generated dice roll to the screen, for example, if your variables (that store the randomly generated die rolls) are called `die1`, `die2`, `die3`, `die4`, and `die5`, you would then modify the code from stage one so that the randomly generated values are passed to the function instead, like so:

```
# Place code to randomly generate the roll of five dice here...
:
:
dice.display_dice(die1, die2, die3, die4, die5)
```

Sample output (this will look different given we are generating random values here):

```
Die 1      Die 2      Die 3      Die 4      Die 5
[3]        [6]        [4]        [6]        [1]

*          * *        * *        * *
*          * *        *          * *        *
*          * *        * *        * *
```

Stage 3

For each dice rolled, you must calculate the number of petals around the rose. Add code that calculates the number of petals for each dice. You will need to add up the petals for each individual dice (0, 2 or 4) and store the total in a variable in order to compare it to the user's guess later (see Stage 4 below).

Refer to section PART I SPECIFICATION – PETALS AROUND THE ROSE on how to calculate the number of petals. For example, the total number of petals in the sample output given in **Stage 2** above is: *2 petals*

Stage 4

Add code to prompt for and read the user's guess (i.e. the score guess for the roll) and display an appropriate message to the screen depending on the user's input. For example

- If the user guesses correctly, the following message is displayed:

```
Well done! You guessed it!
```

- If the user enters an incorrect non-even number, the following message is displayed to the screen, for example:

```
No sorry, it's 0 not 9. The score is always even.
```

- If the user enters an incorrect even number, the following message is displayed to the screen, for example:

```
No sorry, it's 0 not 6.
```

The numbers in the above output examples will change depending on what the user has entered and the score value for the roll.

Sample output 1:

| Die 1 | Die 2 | Die 3 | Die 4 | Die 5 |
|-------|-------|-------|-------|-------|
| [5] | [4] | [6] | [2] | [4] |
| * * | * * | * * | * | * * |
| * | | * * | | |
| * * | * * | * * | * | * * |

```
Please enter your guess for the roll: 4
```

```
Well done! You guessed it!
```

Display the user's guess, the actual score and appropriate message to the screen as seen below:

Sample output 2:

| Die 1 | Die 2 | Die 3 | Die 4 | Die 5 |
|-------|-------|-------|-------|-------|
| [3] | [6] | [3] | [4] | [1] |
| * | * * | * | * * | |
| * | * * | * | | * |
| * | * * | * | * * | |

Please enter your guess for the roll: 7

No sorry, it's 4 not 7. The score is always even.

Sample output 3:

| Die 1 | Die 2 | Die 3 | Die 4 | Die 5 |
|-------|-------|-------|-------|-------|
| [4] | [5] | [5] | [2] | [6] |
| * * | * * | * * | * | * * |
| | * | * | | * * |
| * * | * * | * * | * | * * |

Please enter your guess for the roll: 4

No sorry, it's 8 not 4.

Stage 5

Now...it's time to allow the player to play more than one game. Let's add a loop that loops until the user either enters 'n' (to quit the game). Think about where this code should go – what needs to be repeated, etc.

Sample output:

Would you like to play Petals Around the Rose [y|n]? y

| Die 1 | Die 2 | Die 3 | Die 4 | Die 5 |
|-------|-------|-------|-------|-------|
| [3] | [2] | [4] | [1] | [5] |
| * | * | * * | | * * |
| * | | | * | * |
| * | * | * * | | * * |

Please enter your guess for the roll: 6

Well done! You guessed it!

Roll dice again [y|n]? y

Sample output continued next page...

| Die 1 | Die 2 | Die 3 | Die 4 | Die 5 |
|-------|-------|-------|-------|-------|
| [6] | [2] | [4] | [3] | [3] |
| * * | * | * * | * | * |
| * * | | | * | * |
| * * | * | * * | * | * |

Please enter your guess for the roll: 3

No sorry, it's 4 not 3. The score is always even.

Roll dice again [y|n]? y

| Die 1 | Die 2 | Die 3 | Die 4 | Die 5 |
|-------|-------|-------|-------|-------|
| [2] | [4] | [6] | [6] | [6] |
| * | * * | * * | * * | * * |
| | | * * | * * | * * |
| * | * * | * * | * * | * * |

Please enter your guess for the roll: 8

No sorry, it's 0 not 8.

Roll dice again [y|n]? n

Stage 6

Add code to keep track of how many games were played, the number of correct guesses and the number of incorrect guesses. Display this to the screen as seen in the sample output.

Stage 7

Add code to keep track of how many correct and incorrect guesses are recorded in a row and display the appropriate messages to the screen (only if four in a row are achieved) as seen in the sample output.

Stage 8

Add code to validate the following user input only:

- Would you like to play Petals Around the Rose [y|n]?

Sample output:

```
Would you like to play Petals Around the Rose [y|n]? w
Please enter either 'y' or 'n'.
```

```
Would you like to play Petals Around the Rose [y|n]? y
```

- Roll dice again [y|n]?

Sample output:

```
Roll dice again [y|n]? z
Please enter either 'y' or 'n'.
```

```
Roll dice again [y|n]? y
```

Stage 9

Add code to keep track of dice roll statistics. That is, how many times each die face value was rolled. Display this information to the screen (as seen in the sample output). You **MUST** use a list in order to store this information. You **MUST** also use nested loops in order to display this information.

To define a list in order to store dice roll stats: `die_count = [0,0,0,0,0,0]`

| | | | | | | |
|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| die_count | | | | | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| die_count[0] | die_count[1] | die_count[2] | die_count[3] | die_count[4] | die_count[5] | die_count[6] |

Given that die face values are 1 – 6 inclusive, we create a list with seven elements but ignore the zero element of the list. This will make it easier to increment the appropriate list element.

To access and update the appropriate list element (using die value as an index):

```
die1 = random.randint(1,6)
die_count[die1] = die_count[die1] + 1
```

For example: If die1 is assigned the value 3.

| | | | | | | |
|--------------|--------------|--------------|---------------------|--------------|--------------|--------------|
| die_count | | | | | | |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| die_count[0] | die_count[1] | die_count[2] | die_count[3] | die_count[4] | die_count[5] | die_count[6] |

Algorithm to display the frequency to the screen:

```
index = 1
WHILE index < length of die_count list
    display die face value to the screen (value of index)

    FOR number in 0 to die_count[index]
        display a star to the screen

    increment index by 1
```

Stage 10

Finally, check the sample output (see section titled 'Sample Output – Part I' towards the end of this document) and if necessary, modify your code so that:

- Your program displays the instructions each time it is run (i.e., not each time a user wishes to reroll the dice but each time you *launch* your program).
- The output produced by your program **EXACTLY** adheres to the sample output provided.
- Your program behaves as described in these specs **and** the sample output provided.

PART II SPECIFICATION – NUMBER CONVERTER

Write a **menu driven program** called `yourSaibtId_converter.py` that will allow the user to enter commands and process these commands until the quit command is entered.

Your program will accept entry of a number (in either decimal or binary notation) from the keyboard and convert it to either binary or decimal notation as requested.

Please note: We will be working with two numeral systems only for this part of the assignment, decimal and binary. Decimal refers to Base 10 and works with ten digits, thus, decimal numbers use digits 0-9. Binary refers to Base 2 and works with two digits, thus, binary numbers use digits 0-1.

The following commands should be allowed:

1) Convert to binary:

Prompt for and read (from the keyboard) a number (decimal whole numbers only) to be converted to binary. Convert the decimal number to the binary representation of the number as a string of 0s and 1s. Display the string (binary number) to the screen.

2) Convert to decimal:

Convert a binary number to a decimal number by prompting for and reading (from the keyboard) a binary number (as a string). Convert the binary number to the decimal representation of the number (integer). Display the decimal number (integer) to the screen.

3) Binary counting:

Prompt for and read a decimal number (whole numbers only). Display all decimal numbers starting from 1 up to and including the decimal number entered along with the binary representation of the numbers to the screen.

4) Quit:

Quits the program displaying a goodbye message to the screen.

PRACTICAL REQUIREMENTS (PART II)

It is recommended that you develop this part of the assignment in the suggested stages. Each stage is worth a portion of the marks.

It is expected that your solution will include the use of:

- Your solution in a file called `yourSaibtId_converter.py`.
- Appropriate and well constructed `while` and/or `for` loops. (Marks will be lost if you use `break` statements in order to exit from loops).
- Appropriate `if`, `if-else`, `if-elif-else` statements (as necessary).
- Output that strictly adheres to the assignment specifications. If you are not sure about these details, you should check with the 'Sample Output – Part II' provided at the end of this document.
- Good programming practice:
 - Consistent commenting, layout and indentation. You are to provide comments to describe: your details, program description, all variable definitions, and every significant section of code.
 - Meaningful variable names.
- Your solutions **MAY** make use of the following built-in functions and methods:
 - Built-in functions `int()`, `input()`, `print()`, `range()`, `pow()`, `len()` and `str()`.
- Your solutions **MAY ALSO** make use of the following:
 - Concatenation (+) operator to create/build new strings.
 - Access the individual elements in a string with an index (one element only). i.e. `string_name[index]`.
 - You must use a loop(s) in your solution.
- Your solutions **MUST NOT** use:
 - Built-in functions (other than the `int()`, `input()`, `print()`, `range()`, `pow()`, `len()` and `str()` functions).
 - The built-in functions `int()` and/or `bin()` in order to convert between number systems, i.e., to perform the conversion from binary to decimal and vice versa.
 - Slice expressions to select a range of elements from a string or list. i.e. `name[start:end]`.
 - String or list methods (other than those used for input validation and the `append()` method. i.e. `list_name.append(item)`).

NOTE: **Do not** use `break`, or `continue` statements in your solution – doing so will result in a significant mark deduction. **Do not** use the `quit()` or `exit()` functions or the `return` statement as a way to break out of loops.

PLEASE NOTE: You are reminded that you should ensure that all input and output conform to the specifications listed here; if you are not sure about these details you should check with the sample output provided at the end of this document or post a message to the discussion forum in order to seek clarification.

Please ensure that you use Python 3.5.1 or a later version (i.e. the latest version) in order to complete your assignments. Your programs **MUST** run using Python 3.5.1 (or latest version).

STAGES (PART II)

It is recommended that you develop this part of the assignment in the suggested stages. Each stage is worth a portion of the marks. Many problems in later stages are due to errors in early stages. **Make sure you have finished and thoroughly tested each stage before continuing.**

The following stages of development are recommended:

Stage 1

Implement the interactive mode (prompt for and read menu commands). Set up a loop to obtain and process commands. Test to ensure that this is working correctly before moving onto stage 2. You need not perform any number conversion at this point, you may simply display an appropriate message to the screen, for example:

Sample output:

```
*** Menu ***

1. Convert to binary
2. Convert to decimal
3. Binary counting
4. Quit

What would you like to do [1,2,3,4]? 1
In command 1 { convert to binary

*** Menu ***

1. Convert to binary
2. Convert to decimal
3. Binary counting
4. Quit

What would you like to do [1,2,3,4]? 2
In command 2 { convert to decimal

*** Menu ***

1. Convert to binary
2. Convert to decimal
3. Binary counting
4. Quit

What would you like to do [1,2,3,4]? 3
In command 3 { binary counting

*** Menu ***

1. Convert to binary
2. Convert to decimal
3. Binary counting
4. Quit

What would you like to do [1,2,3,4]? 4

Goodbye.
```

Make sure the program runs correctly. Once you have that working, back up your program. Note: When developing software, you should always have fixed points in your development where you know your software is bug free and runs correctly.

Stage 2

Add code to implement command 1. **Convert to binary.**

Prompt for and read a number (decimal notation) to be converted. Convert the decimal number to the binary representation of the number as a string of 0s and 1s. Display the string (binary number) to the screen. You **must** use a loop(s) in your solution.

Hint:

To convert a whole decimal number to binary:

- Divide the decimal number by 2 recording the quotient and the remainder resulting from the division.
- Continue to divide each new quotient by 2 until you reach a quotient of 0 (remember that the quotient is the result of division). Write the remainders after each step of division (which will either be 1 or 0) off to the side.
- Since we are dividing by 2, when the dividend is even, the binary remainder will be 0 and when the dividend is odd, the binary remainder will be 1.
- Finally, write down the remainders in reverse order (with the last remainder written first). This result is the converted binary number.

For example, to convert the number 92_{10} to binary:

```
92 / 2 - quotient = 46 remainder = 0
46 / 2 - quotient = 23 remainder = 0
23 / 2 - quotient = 11 remainder = 1
11 / 2 - quotient = 5  remainder = 1
 5 / 2 - quotient = 2  remainder = 1
 2 / 2 - quotient = 1  remainder = 0
 1 / 2 - quotient = 0  remainder = 1
```

$92_{10} = 1011100_2$.

Stage 3

Add code to implement command 2. **Convert to decimal.** Convert a binary number to a decimal number by prompting for and reading (from the keyboard) a binary number (as a string). Convert the binary number to the decimal representation of the number (integer). Display the decimal number (integer) to the screen. You **must** use a loop(s) in your solution.

Hint:

To convert a binary number to decimal, first list the powers of 2 from right to left (and their corresponding decimal values), for example:

| 2_7 | 2_6 | 2_5 | 2_4 | 2_3 | 2_2 | 2_1 | 2_0 |
|-------|-------|-------|-------|-------|-------|-------|-------|
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
| | | | | | | | |

Write the digits of the binary number below their corresponding powers of two. For example, to convert the number 1011100_2 to decimal:

| 2_7 | 2_6 | 2_5 | 2_4 | 2_3 | 2_2 | 2_1 | 2_0 |
|-------|-------|-------|-------|-------|-------|-------|-------|
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
| | 1 | 0 | 1 | 1 | 1 | 0 | 0 |

Finally, add up all of the decimal values of the non-zero bits, i.e.: $64 + 16 + 8 + 4 = 92_{10}$.

Stage 4

Add code to implement command 3. **Binary counting.**

Prompt for and read a decimal number (whole number). Display all decimal numbers starting from 1 up to and including the decimal number entered along with the binary representation of the numbers to the screen. You **must** use nested loop(s) in your solution to achieve this (refer to section SAMPLE OUTPUT – PART II)

Stage 5

Ensure that you validate all user input with an appropriate message being displayed for incorrect input entered by the user. You may like to make use of appropriate string method(s) in order to do this (this is the only place that you may make use of string methods. Hint – check the Python online docs for help with this:

<https://docs.python.org/3/library/stdtypes.html#string-methods>).

Menu input should also be validated with an appropriate message being displayed if an incorrect menu command is entered by the user.

Add code to validate all user input. Hint: use a while loop to validate input.

Stage 6

Finally, check the sample output (see section titled ‘Sample Output – Part II’ towards the end of this document) and if necessary, modify your code so that:

- The output produced by your program EXACTLY adheres to the sample output provided.
- Your program behaves as described in these specs and the sample output provided.

SUBMISSION DETAILS

You are required to demonstrate your assignment to your practical supervisor during your week 8 class for marking. The supervisor will mark your work using the marking criteria included in this document. You MUST attend the practical session that you have been attending all study period in order to have your assignment marked.

You are also required to submit an electronic copy of your program via Moodle. Assignments submitted to Moodle, but not demonstrated during your allocated practical session, will NOT be marked. Likewise, assignments that have been demonstrated during the practical session, but have not been submitted via Moodle, will NOT be marked. Assignments are submitted to Moodle in order to check for plagiarism.

All students must follow the submission instructions below:

- Ensure that your files are named correctly (as per instructions outlined in this document).

- Ensure that the following files are included in your submission:

- `yourSAIBTId_game.py`
- `yourSAIBTId_convertor.py`

- For example:

- `bonjd1402_game.py`
- `bonjd1402_encryptor.py`

All files that you submit must include the following comments.

```
#  
# File: fileName.py  
# Author: your name  
# Saibt Id: your saibt id  
# Description: Assignment 1 { place assignment description here...  
# This is my own work as defined by the University's  
# Academic Misconduct policy.  
#
```

Assignments that do not contain these details may not be marked.

You must submit your program **at the start of class before** you demonstrate the work to your marker. You will also be required to demonstrate that you have correctly submitted your work to Moodle. Work that has not been correctly submitted to Moodle will not be marked.

It is expected that students will make copies of all assignments and be able to provide these if required.

EXTENSIONS AND LATE SUBMISSIONS

There will be no extensions/late submissions for this course without one of the following exceptions:

1. A medical certificate is provided that has the timing and duration of the illness and an opinion on how much the student's ability to perform has been compromised by the illness. Please note if this information is not provided the medical certificate WILL NOT BE ACCEPTED. Late assessment items will not be accepted unless a medical certificate is presented to the Course Coordinator. The certificate must be produced as soon as possible and must cover the dates during which the assessment was to be attempted. In the case where you have a valid medical certificate, the due date will be extended by the number of days stated on the certificate up to five working days.
2. A SAIBT counsellor contacts the Course Coordinator on your behalf requesting an extension. Normally you would use this if you have events outside your control adversely affecting your course work.
3. Unexpected work commitments. In this case, you will need to attach a letter from your work supervisor with your application stating the impact on your ability to complete your assessment.
4. Military obligations with proof.


Applications for extensions must be lodged via learnonline before the due date of the assignment.

Note: Equipment failure, loss of data, 'Heavy work commitments' or late starting of the course are not sufficient grounds for an extension.

ACADEMIC MISCONDUCT

Deliberate academic misconduct such as plagiarism is subject to penalties. Information about Academic integrity can be found in the "Policies and Procedures" section of the SAIBT website (see <https://www.saibt.sa.edu.au/policies>).

MARKING CRITERIA

| | | | |
|--|-----------------------------------|---|--|
|  | | Assessment feedback | |
| Problem Solving and Programming (COMP 1039) | | Assignment 1 - Weighting: 10% - Due: Week 8 | |
| NAME: | MAX MARK | MARK | COMMENT |
| PRODUCES CORRECT RESULTS (OUTPUT) - PART I | | | |
| Petals Around the Rose ----- The name of the game is 'Petals Around the Rose'. The name of the game is important. The computer will roll five dice and ask you to guess the score for the roll. The score will always be zero or an even number. Your mission, should you choose to accept it, is to work out how the computer calculates the score. If you succeed in working out the secret and guess correctly five times in a row, you become a Potentate of the Rose. Would you like to play. | Instructions: [1 mark] | + | -1 No/incorrect instructions -1 No or incorrect line spacing |
| Would you like to play Petals Around the Rose [y n]? p Please enter either 'y' or 'n'. Would you like to play Petals Around the Rose [y n]? n No worries... another time perhaps... :) | Start game: [1 mark] | | -1 No incorrect prompt -1 No or incorrect validation for prompt -1 No or 'No worries' message |
| Die 1 Die 2 Die 3 Die 4 Die 5 [4] [3] [2] [5] [6] * Please enter your guess for the roll: | Display dice: [1 mark] | + | -1 No or incorrect dice display -1 No incorrect prompt |
| Well done! You guessed it! No sorry, it's 0 not 9. The score is always even. No sorry, it's 0 not 6. | Result: [2 marks] | + | -1 No or incorrect win message -1 No/incorrect /lose message |
| Roll dice again [y n]? y Roll dice again [y n]? n Roll dice again [y n]? z Please enter either 'y' or 'n'. | Play again: [1 mark] | + | -1 No or incorrect prompt display -1 No or incorrect validation |
| Game Summary ===== You played 2 games: --> Number of correct guesses: 1 --> Number of incorrect guesses: 1 Dice Roll Stats: Face Frequency 1 * 2 *** 3 * 4 * 5 ** 6 ** Thanks for playing! | Game Summary: [4 marks] | | -1 No or incorrect decoration -1 No or incorrect line spacing -1 No or incorrect game totals -1 No or incorrect dice roll stats |
| Use of randint(1,10) for die simulation While loop for play again (playAgain == 'y') Appropriate if statements | Adhere to Specifications: | - | -1 No or incorrect use of randint(1,10) -1 No or incorrect while -1 No or incorrect if statements -1 For using break statements to exit |
| Comments (your details, prog. description, all variable definitions & code), code layout, meaningful variable names. | Style: | - | -1 Insufficient comments -1 Inconsistent code layout -1 Non-descriptive variable names |
| PART I - TOTAL | 10 | | |

| PRODUCES CORRECT RESULTS (OUTPUT) - PART II | | | |
|--|--|---|--|
| <p>*** Menu *** 1 Convert to binary 2 Convert to decimal 3 Binary counting 4 Quit</p> <p>What would you like to [1,2,3,4]: -1 Invalid choice, please enter either 1, 2, 3 or 4.</p> | <p>Menu: [2 marks]</p> | + | <input type="checkbox"/> -1 No or incorrect line spacing <input type="checkbox"/> -1 No or incorrect menu display <input type="checkbox"/> -1 No or incorrect prompt [1,2,3,4] <input type="checkbox"/> -1 No validation of menu input |
| <p>What would you like to do [1,2,3,4]: 1 Please enter number: nine Please make sure your number contains digits 0-9 only.</p> | <p>Convert to binary: [2 marks]</p> | + | <input type="checkbox"/> -1 No or incorrect binary translation <input type="checkbox"/> -1 No validation of menu input |
| <p>What would you like to do [1,2,3,4]: 2 Please enter binary number: 1090 Please make sure your number contains digits 0-1 only.</p> | <p>Convert to decimal: [2 marks]</p> | + | <input type="checkbox"/> -1 No or incorrect decimal translation <input type="checkbox"/> -1 No validation of menu input |
| <p>What would you like to do [1,2,3,4]? 3 please enter number: five Please make sure your number contains digits 0-9 only.</p> | <p>Binary counting: [3 marks]</p> | + | <input type="checkbox"/> -1 No or incorrect output <input type="checkbox"/> -1 No validation of input |
| <p>Enter an option [1,2,3,4]: 4 Goodbye.</p> | <p>Finish: [1 mark]</p> | + | <input type="checkbox"/> -1 No or incorrect goodbye message |
| <p>While/for loop for calculating binary/decimal number While loop for menu/prompt (choice != 4) Appropriate if statements (in general)</p> | <p>Adheres to Specification:</p> | - | <input type="checkbox"/> -1 No or incorrect loop <input type="checkbox"/> -1 No or incorrect loop <input type="checkbox"/> -1 No or incorrect if statements <input type="checkbox"/> -1 For using break/return statements to exit loops |
| <p>Comments (your details, prog. description, all variable definitions & code), code layout, meaningful variable names.</p> | <p>Style:</p> | - | <input type="checkbox"/> -1 Insufficient comments <input type="checkbox"/> -1 Inconsistent code layout <input type="checkbox"/> -1 Non-descriptive variable names |
| PART II - TOTAL | 10 | | |
| BONUS | | | |
| TOTAL | 20 MARKS | | |
| <p>Comments:</p> | | | |

SAMPLE OUTPUT – PART I

Sample output 1:

Petals Around the Rose

The name of the game is 'Petals Around the Rose'. The name of the game is important. The computer will roll five dice and ask you to guess the score for the roll. The score will always be zero or an even number. Your mission, should you choose to accept it, is to work out how the computer calculates the score. If you succeed in working out the secret and guess correctly five times in a row, you become a Potentate of the Rose.

Would you like to play Petals Around the Rose [y|n]? n

No worries... another time perhaps... :)

Sample output 2:

Petals Around the Rose

The name of the game is 'Petals Around the Rose'. The name of the game is important. The computer will roll five dice and ask you to guess the score for the roll. The score will always be zero or an even number. Your mission, should you choose to accept it, is to work out how the computer calculates the score. If you succeed in working out the secret and guess correctly five times in a row, you become a Potentate of the Rose.

Would you like to play Petals Around the Rose [y|n]? y

| Die 1 | Die 2 | Die 3 | Die 4 | Die 5 |
|-------|-------|-------|-------|-------|
| [2] | [1] | [6] | [5] | [2] |
| * | | * * | * * | * |
| | * | * * | * | |
| * | | * * | * * | * |

Please enter your guess for the roll: 4

Well done! You guessed it!

Roll dice again [y|n]? n

Sample output continued next page...

Game Summary

=====

You played 1 games:

```
|--> Number of correct guesses:      1
|--> Number of incorrect guesses:    0
```

Dice Roll Stats:

| Face | Frequency |
|------|-----------|
|------|-----------|

| | |
|---|----|
| 1 | * |
| 2 | ** |
| 3 | |
| 4 | |
| 5 | * |
| 6 | * |

Thanks for playing!

Sample output 3:

Petals Around the Rose

The name of the game is 'Petals Around the Rose'. The name of the game is important. The computer will roll five dice and ask you to guess the score for the roll. The score will always be zero or an even number. Your mission, should you choose to accept it, is to work out how the computer calculates the score. If you succeed in working out the secret and guess correctly five times in a row, you become a Potentate of the Rose.

Would you like to play Petals Around the Rose [y|n]? y

| Die 1 | Die 2 | Die 3 | Die 4 | Die 5 |
|-------|-------|-------|-------|-------|
| [1] | [6] | [2] | [6] | [2] |
| | * * | * | * * | * |
| * | * * | | * * | |
| | * * | * | * * | * |

Please enter your guess for the roll: 3

No sorry, it's 0 not 3. The score is always even.

Sample output continued next page...

Roll dice again [y|n]? y

| Die 1 | Die 2 | Die 3 | Die 4 | Die 5 |
|-------|-------|-------|-------|-------|
| [3] | [4] | [3] | [3] | [3] |
| * | * * | * | * | * |
| * | | * | * | * |
| * | * * | * | * | * |

Please enter your guess for the roll: 8

Well done! You guessed it!

Roll dice again [y|n]? y

| Die 1 | Die 2 | Die 3 | Die 4 | Die 5 |
|-------|-------|-------|-------|-------|
| [2] | [1] | [5] | [1] | [4] |
| * | | * * | | * * |
| | * | * | * | |
| * | | * * | | * * |

Please enter your guess for the roll: 2

No sorry, it's 4 not 2.

Roll dice again [y|n]? n

Game Summary

=====

You played 3 games:

| | |
|----------------------------------|---|
| --> Number of correct guesses: | 1 |
| --> Number of incorrect guesses: | 2 |

Dice Roll Stats:

| Face | Frequency |
|------|-----------|
| 1 | *** |
| 2 | *** |
| 3 | **** |
| 4 | ** |
| 5 | * |
| 6 | ** |

Thanks for playing!

Sample output 4:

Petals Around the Rose

The name of the game is 'Petals Around the Rose'. The name of the game is important. The computer will roll five dice and ask you to guess the score for the roll. The score will always be zero or an even number. Your mission, should you choose to accept it, is to work out how the computer calculates the score. If you succeed in working out the secret and guess correctly five times in a row, you become a Potentate of the Rose.

Would you like to play Petals Around the Rose [y|n]? y

| Die 1 | Die 2 | Die 3 | Die 4 | Die 5 |
|-------|-------|-------|-------|-------|
| [3] | [3] | [5] | [6] | [3] |
| * | * | * * | * * | * |
| * | * | * | * * | * |
| * | * | * * | * * | * |

Please enter your guess for the roll: 3

No sorry, it's 10 not 3. The score is always even.

Roll dice again [y|n]? y

| Die 1 | Die 2 | Die 3 | Die 4 | Die 5 |
|-------|-------|-------|-------|-------|
| [2] | [1] | [6] | [6] | [3] |
| * | | * * | * * | * |
| | * | * * | * * | * |
| * | | * * | * * | * |

Please enter your guess for the roll: 4

No sorry, it's 2 not 4.

Sample output continued next page...

Roll dice again [y|n]? y

| Die 1 | Die 2 | Die 3 | Die 4 | Die 5 |
|-------|-------|-------|-------|-------|
| [3] | [1] | [5] | [2] | [1] |
| * | | * * | * | |
| * | * | * | | * |
| * | | * * | * | |

Please enter your guess for the roll: 7

No sorry, it's 6 not 7. The score is always even.

Roll dice again [y|n]? y

| Die 1 | Die 2 | Die 3 | Die 4 | Die 5 |
|-------|-------|-------|-------|-------|
| [4] | [4] | [5] | [3] | [3] |
| * * | * * | * * | * | * |
| | | * | * | * |
| * * | * * | * * | * | * |

Please enter your guess for the roll: 6

No sorry, it's 8 not 6.

Roll dice again [y|n]? y

| Die 1 | Die 2 | Die 3 | Die 4 | Die 5 |
|-------|-------|-------|-------|-------|
| [5] | [1] | [6] | [3] | [1] |
| * * | | * * | * | |
| * | * | * * | * | * |
| * * | | * * | * | |

Please enter your guess for the roll: 10

No sorry, it's 6 not 10.

Sample output continued next page...

Hint: The name of the game is important... Petals Around the Rose.

Roll dice again [y|n]? y

| Die 1 | Die 2 | Die 3 | Die 4 | Die 5 |
|-------|-------|-------|-------|-------|
| [2] | [3] | [1] | [6] | [5] |
| * | * | | * * | * * |
| | * | * | * * | * |
| * | * | | * * | * * |

Please enter your guess for the roll: 6

Well done! You guessed it!

Roll dice again [y|n]? y

| Die 1 | Die 2 | Die 3 | Die 4 | Die 5 |
|-------|-------|-------|-------|-------|
| [1] | [2] | [4] | [6] | [5] |
| | * | * * | * * | * * |
| * | | | * * | * |
| | * | * * | * * | * * |

Please enter your guess for the roll: 12

No sorry, it's 4 not 12.

Roll dice again [y|n]? n

Game Summary

=====

You played 7 games:

| | |
|----------------------------------|---|
| --> Number of correct guesses: | 1 |
| --> Number of incorrect guesses: | 6 |

Sample output continued next page...

Dice Roll Stats:

| Face | Frequency |
|------|-----------|
| 1 | ***** |
| 2 | **** |
| 3 | ***** |
| 4 | *** |
| 5 | ***** |
| 6 | ***** |

Thanks for playing!

Sample output 5:

Petals Around the Rose

The name of the game is 'Petals Around the Rose'. The name of the game is important. The computer will roll five dice and ask you to guess the score for the roll. The score will always be zero or an even number. Your mission, should you choose to accept it, is to work out how the computer calculates the score. If you succeed in working out the secret and guess correctly five times in a row, you become a Potentate of the Rose.

Would you like to play Petals Around the Rose [y|n]? y

| Die 1 | Die 2 | Die 3 | Die 4 | Die 5 |
|-------|-------|-------|-------|-------|
| [3] | [4] | [1] | [6] | [5] |
| * | * * | | * * | * * |
| * | | * | * * | * |
| * | * * | | * * | * * |

Please enter your guess for the roll: 6

Well done! You guessed it!

Roll dice again [y|n]? y

| Die 1 | Die 2 | Die 3 | Die 4 | Die 5 |
|-------|-------|-------|-------|-------|
| [3] | [5] | [6] | [3] | [2] |
| * | * * | * * | * | * |
| * | * | * * | * | |
| * | * * | * * | * | * |

Sample output continued next page...

Please enter your guess for the roll: 3

No sorry, it's 8 not 3. The score is always even.

Roll dice again [y|n]? y

| Die 1 | Die 2 | Die 3 | Die 4 | Die 5 |
|-------|-------|-------|-------|-------|
| [5] | [4] | [5] | [2] | [3] |
| * * | * * | * * | * | * |
| * | | * | | * |
| * * | * * | * * | * | * |

Please enter your guess for the roll: 10

Well done! You guessed it!

Roll dice again [y|n]? y

| Die 1 | Die 2 | Die 3 | Die 4 | Die 5 |
|-------|-------|-------|-------|-------|
| [5] | [3] | [5] | [2] | [3] |
| * * | * | * * | * | * |
| * | * | * | | * |
| * * | * | * * | * | * |

Please enter your guess for the roll: 12

Well done! You guessed it!

Roll dice again [y|n]? y

| Die 1 | Die 2 | Die 3 | Die 4 | Die 5 |
|-------|-------|-------|-------|-------|
| [5] | [2] | [6] | [4] | [5] |
| * * | * | * * | * * | * * |
| * | | * * | | * |
| * * | * | * * | * * | * * |

Sample output continued next page...

Please enter your guess for the roll: 8

Well done! You guessed it!

Roll dice again [y|n]? y

| Die 1 | Die 2 | Die 3 | Die 4 | Die 5 |
|-------|-------|-------|-------|-------|
| [4] | [3] | [3] | [5] | [2] |
| * * | * | * | * * | * |
| | * | * | * | |
| * * | * | * | * * | * |

Please enter your guess for the roll: 8

Well done! You guessed it!

Roll dice again [y|n]? y

| Die 1 | Die 2 | Die 3 | Die 4 | Die 5 |
|-------|-------|-------|-------|-------|
| [4] | [1] | [2] | [1] | [2] |
| * * | | * | | * |
| | * | | * | |
| * * | | * | | * |

Please enter your guess for the roll: 0

Well done! You guessed it!

Congratulations! You have worked out the secret!
Make sure you don't tell anyone!

Roll dice again [y|n]? y

| Die 1 | Die 2 | Die 3 | Die 4 | Die 5 |
|-------|-------|-------|-------|-------|
| [5] | [5] | [3] | [4] | [1] |
| * * | * * | * | * * | |
| * | * | * | | * |
| * * | * * | * | * * | |

Sample output continued next page...

Please enter your guess for the roll: 2

No sorry, it's 10 not 2.

Roll dice again [y|n]? y

| Die 1 | Die 2 | Die 3 | Die 4 | Die 5 |
|-------|-------|-------|-------|-------|
| [2] | [1] | [6] | [5] | [6] |
| * | | * * | * * | * * |
| | * | * * | * | * * |
| * | | * * | * * | * * |

Please enter your guess for the roll: 4

Well done! You guessed it!

Roll dice again [y|n]? n

Game Summary

=====

You played 9 games:

| | |
|----------------------------------|---|
| --> Number of correct guesses: | 7 |
| --> Number of incorrect guesses: | 2 |

Dice Roll Stats:

| Face | Frequency |
|------|-----------|
| 1 | ***** |
| 2 | ***** |
| 3 | ***** |
| 4 | ***** |
| 5 | ***** |
| 6 | ***** |

Thanks for playing!

Sample output 6:

Petals Around the Rose

The name of the game is 'Petals Around the Rose'. The name of the game is important. The computer will roll five dice and ask you to guess the score for the roll. The score will always be zero or an even number. Your mission, should you choose to accept it, is to work out how the computer calculates the score. If you succeed in working out the secret and guess correctly five times in a row, you become a Potentate of the Rose.

Would you like to play Petals Around the Rose [y|n]? p
Please enter either 'y' or 'n'.

Would you like to play Petals Around the Rose [y|n]? y

| Die 1 | Die 2 | Die 3 | Die 4 | Die 5 |
|-------|-------|-------|-------|-------|
| [4] | [3] | [4] | [1] | [1] |
| * * | * | * * | | |
| | * | | * | * |
| * * | * | * * | | |

Please enter your guess for the roll: 2

Well done! You guessed it!

Roll dice again [y|n]? z
Please enter either 'y' or 'n'.

Roll dice again [y|n]? n

Game Summary

=====

You played 1 games:

| | |
|----------------------------------|---|
| --> Number of correct guesses: | 1 |
| --> Number of incorrect guesses: | 0 |

Sample output continued next page...

Dice Roll Stats:

| Face | Frequency |
|------|-----------|
| 1 | ** |
| 2 | |
| 3 | * |
| 4 | ** |
| 5 | |
| 6 | |

Thanks for playing!

SAMPLE OUTPUT – PART II

Sample output 1:

*** Menu ***

1. Convert to binary
2. Convert to decimal
3. Binary counting
4. Quit

What would you like to do [1,2,3,4]? 1

Please enter number: 92

Binary number: 1011100

*** Menu ***

1. Convert to binary
2. Convert to decimal
3. Binary counting
4. Quit

What would you like to do [1,2,3,4]? 1

Please enter number: 3

Binary number: 11

*** Menu ***

1. Convert to binary
2. Convert to decimal
3. Binary counting
4. Quit

What would you like to do [1,2,3,4]? 1

Please enter number: 7

Binary number: 111

Sample output continued next page...

*** Menu ***

1. Convert to binary
2. Convert to decimal
3. Binary counting
4. Quit

What would you like to do [1,2,3,4]? 1

Please enter number: 10

Binary number: 1010

*** Menu ***

1. Convert to binary
2. Convert to decimal
3. Binary counting
4. Quit

What would you like to do [1,2,3,4]? 2

Please enter binary number: 10

Decimal number: 2

*** Menu ***

1. Convert to binary
2. Convert to decimal
3. Binary counting
4. Quit

What would you like to do [1,2,3,4]? 2

Please enter binary number: 11

Decimal number: 3

*** Menu ***

1. Convert to binary
2. Convert to decimal
3. Binary counting
4. Quit

What would you like to do [1,2,3,4]? 2

Please enter binary number: 1010

Decimal number: 10

Sample output continued next page...

*** Menu ***

1. Convert to binary
2. Convert to decimal
3. Binary counting
4. Quit

What would you like to do [1,2,3,4]? 2

Please enter binary number: 111

Decimal number: 7

*** Menu ***

1. Convert to binary
2. Convert to decimal
3. Binary counting
4. Quit

What would you like to do [1,2,3,4]? 2

Please enter binary number: 1011100

Decimal number: 92

*** Menu ***

1. Convert to binary
2. Convert to decimal
3. Binary counting
4. Quit

What would you like to do [1,2,3,4]? 3

Please enter number: 10

Decimal: 1 = binary: 1
Decimal: 2 = binary: 10
Decimal: 3 = binary: 11
Decimal: 4 = binary: 100
Decimal: 5 = binary: 101
Decimal: 6 = binary: 110
Decimal: 7 = binary: 111
Decimal: 8 = binary: 1000
Decimal: 9 = binary: 1001
Decimal: 10 = binary: 1010

Sample output continued next page...

*** Menu ***

1. Convert to binary
2. Convert to decimal
3. Binary counting
4. Quit

What would you like to do [1,2,3,4]? 4

Goodbye.

Sample output 2:

*** Menu ***

1. Convert to binary
2. Convert to decimal
3. Binary counting
4. Quit

What would you like to do [1,2,3,4]? 9

Invalid choice, please enter either 1, 2, 3 or 4.

What would you like to do [1,2,3,4]? 1

Please enter number: nine

Please make sure your number contains digits 0-9 only.

Please enter number: 2t

Please make sure your number contains digits 0-9 only.

Please enter number: 8

Binary number: 1000

*** Menu ***

1. Convert to binary
2. Convert to decimal
3. Binary counting
4. Quit

What would you like to do [1,2,3,4]? 2

Please enter binary number: 1090

Please make sure your number contains digits 0-1 only.

Please enter binary number: 11j100

Please make sure your number contains digits 0-1 only.

Please enter binary number: 1011

Decimal number: 11

Sample output continued next page...

*** Menu ***

1. Convert to binary
2. Convert to decimal
3. Binary counting
4. Quit

What would you like to do [1,2,3,4]? 3

Please enter number: five

Please make sure your number contains digits 0-9 only.

Please enter number: 5

Decimal: 1 = binary: 1
Decimal: 2 = binary: 10
Decimal: 3 = binary: 11
Decimal: 4 = binary: 100
Decimal: 5 = binary: 101

*** Menu ***

1. Convert to binary
2. Convert to decimal
3. Binary counting
4. Quit

What would you like to do [1,2,3,4]? 4

Goodbye.