Lab 1: Expressions, Statements, and Control structures

Name: _____ University Number: _____

Exercise 1: Surface Area and Volume of a Hexagonal Pyramid

AIM:

Write a Python program that prompts the user for side length *l* of the base and height *h* in cm of a hexagonal pyramid (i.e. a pyramid with a hexagonal base and six isosceles triangular faces that intersect at the apex), computes the surface area *A* and volume *V* of the pyramid using the formulas $A = 3l(\sqrt{3}l + \sqrt{3l^2 + 4h^2})/2$ and $V = \sqrt{3}hl^2/2$, and finally outputs the results. Here are the sample input and output of this program:

```
Enter the side length l of the base of the pyramid in cm: 3
Enter the height h of the pyramid in cm: 4
The surface area of the pyramid is 66.3099499659424 cm^2.
The volume of the pyramid is 31.17691453623979 cm^3.
```

ALGORITHM:

1. Start
2. Import the square root function from the math module
3. Input the side length of the base and height of a hexagonal pyramid
4. Calculate the surface area and volume of the pyramid
5. Output the results
6. Stop

PROGRAM:

```
# Exercise 1: Surface Area and Volume of a Hexagonal Pyramid
# Written by F K Chow, HKU
# Latest update: 2021/12/9

# Import the square root function from the math module
from math import sqrt

# Input the side length of the base and height of the pyramid
```

```
l = float(input('Enter the side length l of the base of the pyramid \
in cm: '))
h = float(input('Enter the height h of the pyramid in cm: '))

# Compute the surface area and the volume of the cone
A = 3*l*(sqrt(3.0)*l + sqrt(3*l**2+4*h**2))/2.0
V = sqrt(3.0)*h*(l**2)/2

# Display the computing results
print('The surface area of the pyramid is', A, 'cm^2.')
print('The volume of the pyramid is', V, 'cm^3.')
```

OUTPUT:

```
Enter the side length l of the base of the pyramid in cm: 6
Enter the height h of the pyramid in cm: 5
The surface area of the pyramid is 223.330589525423 cm^2.
The volume of the pyramid is 155.88457268119893 cm^3.
```

Exercise 2: Operation of Two Integers

AIM:

Write a Python program that prompts for two integer operands and one of the binary operators +, –, *, /, or % from the user, performs the operation on the operands with the operator using an `if-elif-else` statement, and finally prints the result. Your program should check whether the input operator is valid. Here are the sample input and output of this program:

```
Enter the integer A: 5
Enter the integer B: 4
Enter a binary operator (+,-,*,/,%): +
5 + 4 = 9
```

```
Enter the integer A: 2
Enter the integer B: 3
Enter a binary operator (+,-,*,/,%): =
Invalid input. The operator must be one of the followings: +,-,*,/,%.
```

ALGORITHM:

1. Start
2. Input the integer operands and binary operator
3. Perform the operation if the input operator is valid
4. Output the result
5. Stop

PROGRAM:

```
# Exercise 2: Operation of Two Integers
# Written by F K Chow, HKU
# Last update: 2021/12/9

# Input the integer operands and binary operator
A = int(input('Enter the integer A: '))
B = int(input('Enter the integer B: '))
bo = input('Enter a binary operator (+,-,*,/,%): ')
```

```
# Perform the operation if the input operator is valid
# And then output the result
if bo == '+':
    print(A, '+', B, '=', A+B)
elif bo == '-':
    print(A, '-', B, '=', A-B)
elif bo == '*':
    print(A, '*', B, '=', A*B)
elif bo == '/':
    print(A, '/', B, '=', A/B)
elif bo == '%':
    print(A, '%', B, '=', A%B)
else:
    print('Invalid input. The operator must be one of the followings:',
          ' +,-,*,/,%.')
```

OUTPUT:

```
Enter the integer A: 9
Enter the integer B: 5
Enter a binary operator (+,-,*,/,%): %
9 % 5 = 4

Enter the integer A: 7
Enter the integer B: 4
Enter a binary operator (+,-,*,/,%): &
Invalid input. The operator must be one of the followings: +,-,*,/,%.
```

Exercise 3: Taylor Series of $x\,(1+x^2)^{1/2}$

AIM:

Write a Python program that prompts for a real number x in the open interval (–1, 1) and a positive integer n from the user, computes the sum of the first n terms of the Taylor series

$$x\sqrt{1+x^2} = \sum_{i=0}^{n-1} \frac{(-1)^{i-1}(2i)!}{4^i (i!)^2 (2i-1)} x^{2i+1}$$

for any $x \in (-1, 1)$ using a `for` loop, and finally output the result. Your program should check whether the user input is valid. Here are the sample input and output of this program:

```
Enter a real number x in (-1, 1): 0.6
Enter a positive integer n: 5
The sum of first 5 terms of the Taylor series of x*(1+x^2)^(1/2)
for x = 0.6 is 0.6996359400000001
Enter a real number x in (-1, 1): 2.2
Enter a positive integer n: 7
Invalid input. x must have absolute value less than 1!
Enter a real number x in (-1, 1): -0.9
Enter a positive integer n: -5
Invalid input. n must be a positive integer!
```

ALGORITHM:

1. Start
2. Import the factorial function from the math module
3. Input a real number x in (–1, 1) and a positive integer n
4. If the user input is invalid, then output an error message and jump to step 7
5. Initialize the sum to zero. For i = 1, 2, … n–1, calculate the ith term of the Taylor series of $x\sqrt{1+x^2}$ and add it to the sum.
6. Output the sum as the result
7. Stop

(The program is shown on the next page.)

PROGRAM:

```
# Exercise 3: Taylor Series of x*(1+x^2)^(1/2)
# Written by F K Chow, HKU
# Last update: 2022/1/26

# Import the factorial function from the math module
from math import factorial

# Input a real number x with absolute value less than 1 and a positive
# integer n
x = float(input('Enter a real number x in (-1, 1): '))
n = int(input('Enter a positive integer n: '))

# Ensure that the user input is valid
if abs(x) >= 1:
    print('Invalid input. x must have absolute value less than 1!')
elif n <= 0:
    print('Invalid input. n must be a positive integer!')
else:
    # Compute the sum of the first n terms of the Taylor series of
    # x*(1+x^2)^(1/2)
    sum = 0
    for i in range(n):
        coeff = (-1)**(i-1)*factorial(2*i)/(4**i*factorial(i)**2 \
                                            *(2*i-1))
        sum += coeff*x**(2*i+1)

    # Output the result
    print('The sum of first', n, 'terms of the Taylor series of '
          'x*(1+x^2)^(1/2) ')
    print('for x =', x, 'is', sum)
```

(The output is shown on the next page.)

OUTPUT:

```
Enter a real number x in (-1, 1): -0.54
Enter a positive integer n: 3
The sum of first 3 terms of the Taylor series of x*(1+x^2)^(1/2)
for x = -0.54 is -0.6129924372000001

Enter a real number x in (-1, 1): -1.6
Enter a positive integer n: 6
Invalid input. x must have absolute value less than 1!

Enter a real number x in (-1, 1): 0.8
Enter a positive integer n: -4
Invalid input. n must be a positive integer!
```

Exercise 4: Greatest Common Divisor

AIM:

The Greatest Common Divisor (GCD) of two integers is the largest positive integer that divides both of them without leaving a remainder. An efficient way for finding the GCD of two natural numbers (i.e. non-negative integers) is the Euclidean algorithm which works as follows:

(a) If one of the numbers is zero, then the GCD is the other non-zero number and we can stop.

(b) Compute the remainder of the larger number divided by the smaller one and then replace the larger number by the remainder.

(c) Repeat step (b) until the remainder is zero. The GCD is the larger number among the numbers after the replacement in (b).

Write a Python program that prompts the user for two natural numbers, find their GCD with the Euclidean algorithm using a `while` loop, an `if-else` statement, and `if-elif-else` statements, and finally outputs the result. Your program should check whether the user input is valid. Here are the sample input and output of this program:

```
Enter a natural number x: 168
Enter another natural number y: 180
The GCD of 168 and 180 is 12
Enter a natural number x: 25
Enter another natural number y: -10
Invalid input. Both x and y must be integers >= 0!
Enter a natural number x: 18
Enter another natural number y: 18
Invalid input. x and y must be different numbers!
```

ALGORITHM:

1. Start
2. Input two natural numbers x and y
3. If $x < 0$ or $y < 0$ or $x = y$, then output an error message and jump to step 6
4. Find the GCD of x and y using the Euclidean algorithm as follows:
   (a) If $x = 0$ or $y = 0$, then set the GCD to be the other non-zero number and jump to step 5.
   (b) If $x > y$, set $a = x$ and $b = y$; otherwise, set $a = y$ and $b = x$. Next, do the followings:
      (i) Compute the remainder $r = a \% b$ and then set $a = b$ and $b = r$

(ii) If $r = 0$, set the GCD to be a and jump to step 5; otherwise, back to part (i).

5. Output the GCD found in step 4 as the result
6. Stop

PROGRAM:

```
# Exercise 4: Greatest Common Divisor
# Written by F K Chow, HKU
# Last update: 2022/1/26

# Input two natural numbers x and y
x = int(input('Enter a natural number x: '))
y = int(input('Enter another natural number y: '))

# Ensure that the user input is valid
if (x < 0) or (y < 0):
    print('Invalid input. Both x and y must be integers >= 0!')
elif x == y:
    print('Invalid input. x and y must be different numbers!')
else:
    # Find the GCD of x and y using the Euclidean algorithm
    if x == 0:
        gcd = y
    elif y == 0:
        gcd = x
    else:
        # First set a and b to be the larger and smaller numbers
        # among x and y respectively
        if x > y:
            a = x
            b = y
        else:
            a = y
            b = x
        while True:
            r = a % b
```

```
            a = b
            b = r
            if r == 0:
                gcd = a
                break

    # Output the result
    print('The GCD of', x, 'and', y, 'is', gcd)
```

OUTPUT:

```
Enter a natural number x: 3915
Enter another natural number y: 825
The GCD of 3915 and 825 is 15

Enter a natural number x: -252
Enter another natural number y: 1024
Invalid input. Both x and y must be integers >= 0!

Enter a natural number x: 625
Enter another natural number y: 625
Invalid input. x and y must be different numbers!
```

Exercise 5: Displaying a Number Pattern

AIM:

Write a Python program that prompts the user for a line number n which is a positive integer < 10 and then displays a number pattern of $2n-1$ lines with the following format:

1
121
12321
1234321
123454321
1234321
12321
121
1

using nested `for` loops. Your program should check whether the input value of n is valid.

ALGORITHM:

1. Start
2 Input a line number n
3. If $n < 0$ or $n > 10$, then output an error message and jump to step 5
4 Do the followings for $i = 1, \ldots, 2n - 1$:
   (a) Set $r = i$ if $i \leq n$; otherwise set $r = 2n - i$
   (b) Print $n - r$ spaces
   (c) Print j for $j = 1, 2, \ldots, r$
   (d) Print j for $j = r - 1, \ldots, 2, 1$ and then jump to the next line
5 Stop

PROGRAM:

```
# Exercise 5: Displaying a Number Pattern
# Written by F K Chow, HKU
# Last update: 2021/12/9

# Input a line number n
n = int(input('Enter a line number n (1-9): '))
```

```
# Ensure that the input line number is valid
if (n <= 0) or (n >= 10):
    print('Invalid input. n must be a positive integer < 10!')
# Display the number pattern of 2n-1 lines
else:
    for i in range(1,2*n):
        if i <= n:
            r = i
        else:
            r = 2*n - i
        for j in range(n-r):
            print(' ', end='')
        for j in range(1,r+1):
            print(j, end='')
        for j in range(r-1,0,-1):
            print(j, end='')
        print()
```

OUTPUT:

```
Enter a line number n (1-9): 8
       1
      121
     12321
    1234321
   123454321
  12345654321
 1234567654321
123456787654321
 1234567654321
  12345654321
   123454321
    1234321
     12321
      121
       1

Enter a line number n (1-9): -5
Invalid input. n must be a positive integer < 10!
Enter a line number n (1-9): 13
Invalid input. n must be a positive integer < 10!
```