1. What is the output of this algorithm?

**BEGIN**
 X = 1
 Y = X
 **OUTPUT**(Y)
**END**

1. What are the output(s) of this algorithm?

**BEGIN**
 X = 2
 Y = 10
 **WHILE** X < Y
 **OUTPUT**(X)
 X = X + 2
 **ENDWHILE**
**END**

1. What are the output(s) of this algorithm?

**BEGIN**
 A = 3
 B = A
 C = 1
 **WHILE** B == A
 **IF** C < A
 C = C + 1
 **ELSE**
 A = B + 1
 **ENDIF
 OUTPUT** A, B, C
 **ENDWHILE**
**END**

1. How many times is the word "hello" printed in the following algorithm?

**BEGIN** n = 1
 **WHILE** n < 5
 i = 4
 **WHILE** i > 2
 **IF** i == n **THEN**
 **OUTPUT** "hello"
 **ENDIF**
 i = i - 1
 **ENDWHILE** n = n + 1
 **ENDWHILE
END**

1. What is the output of this algorithm?

**BEGIN**

**BEGIN** *abs*(z)

 **IF** z < 0 **THEN**

 **RETURN** z \* -1

 **ENDIF**

**END**

 **SET** x = 1

 **SET** y = -1

 **IF** *abs*(x \* y) == 1 **THEN**

 **PRINT** "A"

 **ELSE**

 **PRINT** "B"

 **ENDIF**

**END**

1. What is the output of this algorithm?

**BEGIN**

**BEGIN** *unwind*(z,i)

 **IF** i == 1 **THEN**

 **RETURN**(z)

 **ELSE**

 **SET** z = z \* i

 **SET** i = i – 1

 **RETURN** *unwind*(z, i)

 **ENDIF**

**END**

 **SET** x = 2

 **SET** y = 3

 **SET** answer = *unwind*(x,y)

 **PRINT** (answer)

**END**

1. What is the output of this algorithm?

**BEGIN**

**BEGIN** a(i)

 **IF** i < 0 **THEN**

 **RETURN** i \* -2

 **ELSE**

 **RETURN** *i* + 1

 **ENDIF**

**END**

 **SET** result = a(b(-2))

 **PRINT** (result)

**END**

**BEGIN** b(i)

 **RETURN** a(i)

**END**

1. What is the output of this algorithm?

**BEGIN**

**BEGIN** method1(i)

 **RETURN** (i + **24**) % 26

 *//would this return a diff result?:*
 *//RETURN (i – 2) % 26*

**END**

 **SET** list = []
 **SET** counter = 0

 **WHILE** counter < 5
 **SET** list[counter] = method1(counter)
 **SET** counter += 1
 **ENDWHILE** **FOR** item **IN** list
 **PRINT** method2(item)
 **ENDFOR**

**BEGIN** method2(i)

 **RETURN** *chr*(65+i)

 //chr returns string representation of

 //a Unicode integer, *chr*(65) returns “A”

**END**

**END**

1. Desk check the following algorithm recording the OUTPUT of A and B:

|  |  |  |
| --- | --- | --- |
| **BEGIN** Special (X, Y) **IF** X <= 3 **THEN**  **RETURN** X \* 5 **ELSE**  **RETURN** X \* Y **ENDIF****END** Special**BEGIN** **SET** A, B = 1 **PRINT** A, B **WHILE** A < 5 **SET** A = A + 1 **SET** B = Special( A, B ) **PRINT** A, B **ENDWHILE****END** | A | B |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

1. What is the ciphertext created from encrypting the plain\_text\_pin with the key blocks:

**BEGIN**
 plain\_text\_pin = [2,3]
 key\_block\_1 = [1,2,3,4,5,6,7,8,9]
 key\_block\_2 = [9,8,7,6,5,4,3,2,1]
encrypted\_digit\_1 = key\_block\_1[plain\_text\_pin[0]]
 encrypted\_digit\_2 = key\_block\_2[plain\_text\_pin[1]]
 cipher\_text = *str*(encrypted\_digit\_1) + *str*(encrypted\_digit\_2)
**END**

Extension challenges:

1. Re-write the algorithm in question 10 so that it uses an *additional* 2 key blocks to encrypt a 4 digit pin [2,3,4,5]:

|  |  |
| --- | --- |
| key\_block\_3 | [2,4,6,8,1,3,5,7,9] |
| key\_block\_4 | [9,1,8,2,7,3,6,4,5] |

1. Implement the re-written algorithm from question 11 to use a loop mechanism