

Answer Key for Exam A

Preguntas de opción múltiple (5 puntos)

1. [0.5 pt] ¿Qué lista genera la llamada `create_list()`?

```
def create_list():
    i = 0
    lst = []
    while (i <= 10):
        while (j <= 10):
            if (j == i+2):
                lst.append([i, j])
                j = j+1
            i = i+1
    return lst
```

- (a) `[[0,2], [1,3], [2,4], [3,5], [4,6], [5,7], [6,8], [7,9]]`
(b) `[[0,2], [1,3], [2,4], [3,5], [4,6], [5,7], [6,8], [7,9], [8,10]]`
(c) `[[0,2]]`
(d) Ninguna de las opciones anteriores es válida

2. [0.5 pt] ¿Qué valor devolvería la llamada `isPrime(51)`?

```
def isPrime(number):
    top = int(number**0.5)+1
    i = 2
    has_divisor = False
    while i < top and not has_divisor:
        if number%i == 0:
            has_divisor = True
            i += 1
    return not has_divisor
```

- (a) `True`
(b) `False`
(c) `'not has_divisor'`
(d) Nada, el bucle no termina.

3. [0.5 pt] ¿Qué valor devolvería la llamada `check("barco", 5, 'casa')`?

```
def check(a, b, c):
    return (a[5] > c[0]) and (b%3 < 2)
```

- (a) `True`
(b) `False`
(c) `'False'`
(d) Se produce un error en tiempo de ejecución.

4. [0.5 pt] ¿Qué valor devolvería la llamada `highest_square(71)`?

```
def isPerfectSquare(number):
    return (int(number**0.5)**2 == number)

def highest_square(top):
    highest = current = 1
    while current < top:
        if isPerfectSquare(current):
            highest = current
            current += 1
    return highest
```

- (a) `1`
(b) `64`
(c) `81`
(d) Nada, el bucle no termina.

5. [0.5 pt] ¿Qué muestra por pantalla la llamada `func1(3)`?

```
def func1(x):
    a = func2(x//2)
    b = func3(x+1)
    print(a + b)

def func2(x):
    return x**2
    print(x**2)

def func3(y):
    if y < 4:
        print(y**2)
    else:
        print(y**2)
    return 2*y
```

- (a) `1.89`
(b) `8`
(c) `1`
(d) `8.9`

6. [0.5 pt] Suponiendo la siguiente definición:

```
#UK film ratings are as follows: U, PG, 12A, 12, 15, 18, R18
def admit(customer_age, film_rating, cinema_Xlicence, accompanied):
    if customer_age >= 12:
        admission = film_rating
    elif customer_age >= 15:
        admission = film_rating
    elif customer_age >= 18:
        admission = True
    if film_rating == "R18":
        admission = cinema_Xlicence
    else:
        admission = (film_rating == "12A" and accompanied) \
        or (film_rating in ["PG", "U"])
    return admission
```

¿Qué se imprime cuando se ejecuta la siguiente línea de código en el intérprete de Python?
admit(17, "15", False, False), admit(32, "R18", True, False)

- (a) (True, True)
(b) (True, False)
(c) (False, True)
☒ (d) (False, False)

7. ¿Cuál es la salida que produce el fragmento de código siguiente para la llamada `test_if(0)`?

```
def test_if(x):
    if x <= 0:
        print("Value less than or equal to zero")
        x += 1
    if x >= 0:
        print("Value greater than zero")
```

- (a) Value less than or equal to zero
(b) Value greater than zero
(c) Value less than or equal to zero
(d) El código es incorrecto

8. [0.5 pt] ¿Qué valor devolvería la llamada `number_list(3)`?

```
def number_list(top):
    output = []
    for i in range(top):
        for j in range(i, top, -1):
            output.append(j)
    return output
```

- ☒ (a) []
(b) [3, 2, 1, 3, 2, 3]
(c) [3, 3, 2, 3, 2, 1]
(d) [3, 2, 1, 0, -1]

9. ¿Cuál es el valor de `a` tras la instrucción `a = sum_lst([1,2,3,4,5])`?

```
def sum_lst(lst):
    sum = 0
    while i < len(lst):
        sum += lst[i]
        i += 1
    print(sum)
a = sum_lst([1,2,3,4,5])
```

- (a) 15
(b) None
(c) Se produce un error en tiempo de ejecución.
(d) Nada, el bucle no termina.

10. Cual es la salida que produce el fragmento de código siguiente:

```
list = [1,2,3,4,5]
position = 0
i = 0
while i < len(list):
    print(list[position])
    position += 1
```

Dadas las siguientes afirmaciones, ¿cuáles son ciertas?

- 1 El código imprime

0
1
2
3
4

- 2 El código imprime

1
2
3
4
5

- 3 Se produce un error en tiempo de ejecución.

- (a) Sólo la 1
(b) Sólo la 2
(c) La 1 y la 3
(d) La 2 y la 3

Rellena el código que falta (1 punto)

11. [0.5 pt] Rellena los huecos para que la siguiente función devuelva la posición de todas las apariciones de `word` en `text`. Para ello usa el método `find` de las cadenas de caracteres: `find(other, pos)` devuelve la primera aparición posterior o igual a `pos` de `other` en `s`. Si `other` no aparece devuelve `-1`.

```
def all_occurrences(text, word):
    result = []
    last = -----
    while last != -1:
        result.append(last)
        last = -----
    return result
```

12. [0.5 pt] Rellena los huecos para que la siguiente función `exp_of` calcule el mayor `n` tal que existe un `c` tal que $\text{num} = \text{base}^n * c$

```
def exp_of(base, num):
    exp = 0
    while -----:
        num = -----
        exp = -----
    return exp
```

Preguntas para desarrollar (4 puntos)

13. [2 pt] Dada una cadena de caracteres, queremos separar las palabras que las componen. Consideraremos como palabra una sucesión de letras entre caracteres que no sean letras. Por ejemplo, las palabras de "En un lugar, de un, "de", "la", "mancha", "de", "cuyo" y "nombre". Para distinguir si un carácter es una letra o un signo de puntuación o separador puedes suponer definida la siguiente función:

```
import string

def is_letter(c):
    return c not in string.whitespace and c not in string.punctuation
```

Escribe una función en Python que dada una cadena de caracteres devuelva la lista de las palabras que contiene.

14. [2 pt] Como todos sabemos todo número natural num mayor o igual que 1 se puede expresar de la siguiente forma $num = 2^a \cdot 3^b \cdot c$, donde $a, b \geq 0$ y $c \geq 1$ no es divisible ni por 2 ni por 3. Un entero num es pitaguiat si $num \geq 1$ y en las condiciones $a \geq b$. Una lista de enteros será pitaguiat si todos sus números son pitaguiats. Haz un programa que indique si una lista es pitaguiat.