

Facultad de Estudios Estadísticos

Programación I

1er parcial

Fecha: 26 de octubre

Exercise 1: Instrucciones

- No se pueden usar funciones predefinidas ni estructuras de python no vistas en clase.

Exercise 2: Preguntas cortas

Responde de manera concisa a las siguientes preguntas.

Pregunta 1. (1 punto) Considera las funciones

```
def s1(a, b):
    return (a + b) / 2
```

```
def s2(a, b):
    return (a + b) // 2
```

y las llamandas `s1(1, 3)` y `s2(1, 3)`.

a ¿Qué valor devuelve cada llamada?.

b Da una indicación de tipos adecuada para cada función.

Pregunta 2. (1 punto) ¿Qué muestra el siguiente trozo de código?

```
b1 = 10 ** 200 + 1 == 10 ** 200
b2 = 10.0 ** 200 + 1 == 10.0 ** 200
b3 = math.sqrt(2)**2 == 2
b1, b2, b3
```

Pregunta 3. (1 punto) Considera la función

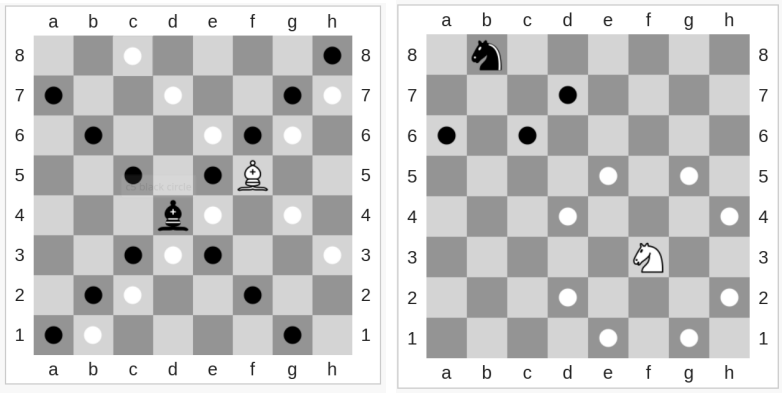
```
def nota2str(nota: float) -> str:
    if nota > 9:
        snota = 'sobresaliente'
    if nota > 7:
        snota = 'notable'
    if nota > 5:
        snota = 'aprobado'
    if nota < 5:
        snota = 'suspenso'
    return snota
```

¿Qué devuelve `nota2str(10)`

Pregunta 4. (1 punto) Da una definición correcta de la función anterior.

Exercise 3: Ajedrez

De entre las figuras del ajedrez, considera el alfil y el caballo.



En este ejercicio vamos a considerar que tanto filas como columnas están numeradas del 1 al 8 (a → 1, b → 2, . . . , h → 8). Describe funciones para las siguientes preguntas.

Pregunta 5. (1 punto) Un movimiento del alfil es correcto.
`can_move_bishop(srow: int, scol: int, drow: int, dcol: int) -> bool`

Pregunta 6. (1 punto) Un movimiento del caballo es correcto.
`can_move_knight(srow: int, scol: int, drow: int, dcol: int) -> bool`

Pregunta 7. (1 punto) Realiza una función
`is_valid(piece: str) -> bool`
que indica si la pieza es válida. Las piezas válidas son: `pawn`, `king`, `bishop`, `rook`, `queen`, `kinght`

Pregunta 8. (1 punto) Realiza una función
`can_move(srow: int, scol: int, drow: int, dcol: int, piece: str) -> bool`
que indica si la pieza puede hace el movimiento, las posibles piezas son las que se indican en el apartado anterior. Debes comprobar que la pieza que se mueve está entre las indicadas. Al igual que están definidas las funciones `can_move_bishop` y `can_move_knight`, debes suponer que alguien ha definido el resto de funciones para cada una de las fichas.

Exercise 4: Sucesiones

Pregunta 9. (1 punto) Una sucesión aritmética en la que cada elemento se consigue del anterior sumando una razón r:

$$ar_0 = a \qquad ar_k = s_{k-1} + r, \quad k > 0$$

Donde a y r son enteros. Define una función
`sum_arit(a: int, r: int, nterm: int) -> int`
que devuelva la suma aritmética de los primeros `nterm` términos de la sucesión. Consideremos s la suma buscada:

$$\begin{array}{rrcccccccc}
 & s = & a & + & a+r & + & a+2r & + & \cdots & + & a+(n-1)r \\
+ & s = & a+(n-1)r & + & a+(n-2)r & + & a+(n-3)r & + & \cdots & + & a \\
\hline
 & 2s = & 2a+(n-1)r & + & 2a+(n-1)r & + & 2a+(n-1)r & + & \cdots & + & 2a+(n-1)r
\end{array}$$

por tanto se cumple $2s = n(2a + (n-1)r)$.

Pregunta 10. (1 punto) Consideremos una sucesión geométrica en la que cada elemento se consigue del anterior multiplicando una razón r:

$$gr_0 = a, \qquad gr_k = gr_{k-1} \cdot r, \quad k > 0$$

donde, en este caso, a y r son números reales. Define una función
`sum_geom(a: float, r: float, nterm: int) -> float`
que sume la sucesión geométrica. En este caso tenemos

$$\begin{array}{rlcl} & s \cdot r & = & a \cdot r^n + a \cdot r^{n-1} + \cdots + a \cdot r \\ - & s & = & -a \cdot r^{n-1} + \cdots - a \cdot r - a \\ \hline & s \cdot (r-1) & = & a \cdot r^n - a \end{array}$$

La función se debe calcular para cualquier valor de r, incluyendo si r toma el valor 1.