1. Se consideran los siguientes puntos para cumplir lo solicitado en la pregunta. En caso de haber seguido otro procedimiento que también entrega la respuesta, se evaluará de todos modos.

   (a) Primero debemos obtener correctamente el punto crítico. Para determinar $x_{\text{mín}}$ debemos resolver la siguiente ecuación, $f'_\alpha(x_{\text{mín}}) = 0$.

   - **[3 puntos]**: Obtener la derivada correctamente,

$$\begin{aligned} f'_\alpha(x) &= \log^\alpha(x) + x\,\alpha\,\log^{\alpha-1}(x)\,\frac{1}{x} \\ &= \log^\alpha(x_)) + \alpha\,\log^{\alpha-1}(x) \end{aligned}$$

   - **[2 puntos]**: Despejar correctamente el punto crítico

$$\begin{aligned} f'_\alpha(x_{\text{mín}}) &= 0 \\ \log^\alpha(x_{\text{mín}}) + \alpha\,\log^{\alpha-1}(x_{\text{mín}}) &= 0 \\ \log^{\alpha-1}(x_{\text{mín}})\,(\log(x_{\text{mín}}) + \alpha) &= 0 \\ \text{Considerando que } \log(x_{\text{mín}}) \neq 0, &\text{ se cancela el término } \log^{\alpha-1}(x_{\text{mín}}). \\ \log(x_{\text{mín}}) + \alpha &= 0 \\ \log(x_{\text{mín}}) &= -\alpha \\ \text{Aplicando la función inversa, en este caso } &\text{la función exponencial.} \\ \exp\left(\log(x_{\text{mín}})\right) &= \exp(-\alpha) \\ x_{\text{mín}} &= \exp(-\alpha) \end{aligned}$$

   Por lo tanto $x_{\text{mín}} = \exp(-\alpha)$ para $f_\alpha(x)$.

   (b)
   - **[10 puntos]**: Construir nueva función $w_\alpha(u)$ para determinar raíz.
   Aplicamos el cambio de variable propuesto en el enunciado $u = \log(x)$ y aplicamos el método de Newton:

$$\begin{aligned} y &= x\,\log^\alpha(x) \\ y &= \exp(u)\,u^\alpha \\ -y &= -u^\alpha\,\exp(u). \end{aligned}$$

   Haciendo las mismas consideraciones del enunciado podemos aplicar la función logaritmo a la expresión anterior. La única consideración adicional es recordar que α es un número entero impar por lo que si u es un número negativo también lo es u^α.

$$\begin{aligned} \log(-y) &= \log(-u^\alpha\,\exp(u)) \\ &= \log(-u^\alpha) + u, \end{aligned}$$

   Luego, nuestra función $w_\alpha(u)$ a la cual debemos encontrar la raíz viene dada por:

$$w_\alpha(u) = \log(-u^\alpha) + u - \log(-y).$$

   - **[5 puntos]**: Obtener la derivada de $w_\alpha(u)$.

$$\begin{aligned} w'_\alpha(u) &= \frac{d}{du}(\log(-u^\alpha) + u - \log(-y)) \\ &= \frac{\alpha\,(-u^{\alpha-1})}{-u^\alpha} + 1 \\ &= \frac{\alpha}{u} + 1 \\ &= \frac{\alpha + u}{u}. \end{aligned}$$

- **[5 puntos]:** Construir la iteración del método de Newton.

$$
\begin{aligned}
u_{i+1} &= u_i - \frac{w_\alpha(u_i)}{w'_\alpha(u_i)} \\
&= u_i - \frac{\log(-u_i^\alpha) + u_i - \log(-y)}{\frac{\alpha + u_i}{u_i}} \\
&= u_i - \frac{u_i}{\alpha + u_i}\left(u_i + \log(-u_i^\alpha) - \log(-y)\right) \\
&= u_i + \frac{u_i}{\alpha + u_i}\left(-u_i - \log(-u_i^\alpha) + \log(-y)\right) \\
&= u_i + \frac{u_i}{\alpha + u_i}\left(\log\left(\frac{y}{u_i^\alpha}\right) - u_i\right)
\end{aligned}
$$

  Notar que si $\alpha = 1$ obtenemos la expresión descrita en el enunciado.

- **[5 puntos]:** Cambio de variable final. Se considera que el punto fijo de la iteración anterior como r, entonces,

$$
\begin{aligned}
r &= \log(x_r), \\
\exp(r) &= x_r.
\end{aligned}
$$

  Por lo tanto, $f_\alpha^{-1}(y) = x_r$.

(c) El código a continuación presenta las componentes principales a evaluar.

```
'''
input:
y   : (double) "y" value.
a   : (integer) "\alpha".
n   : (integer) Max number of iteration to be used.

output:
r   : (double) Root obtained by Newton's method.
'''
def find_inverse_f_alpha(y,a,n):
    u = -10
```

- **[5 puntos]:** Ciclo de `n` iteraciones.

```
for i in np.arange(n):
```

- **[10 puntos]:** Evaluación de el lado derecho de la iteración de punto fijo propuesta.

```
    div = y / np.power(u,a)
    u = u + (u / (a + u))*(np.log(div) - u)
```

- **[10 puntos]:** Cambio de variables final.

```
r = np.exp(u)
return r
```

2. *Visitando la expansión en serie de Taylor de la función exponencial.*

(a)
- **[15 puntos]** La sumatoria se puede representar de la siguiente forma:

$$\begin{aligned}\tau_n(x) &= \sum_{i=0}^{n-1} \underbrace{\gamma_i\, x^i}_{\delta_i} \\ &= \sum_{i=0}^{n-1} \delta_i.\end{aligned}$$

  Donde notamos que lo crucial será sumar los términos δ_i, que es el producto de γ_i y x^i. Sabemos adicionalmente que los δ_i son mayores o iguales que 0. Por lo tanto la forma *adecuada* es sumarlos de menor a mayor para reducir la pérdida de importancia de sumar números con más de 16 órdenes de magnitud de diferencia.
- **[15 puntos]** Ahora se necesita ordenar los números δ_i y luego sumarlos. Por lo tanto si denotamos como $\widehat{\delta}_i$ el resultado de haber ordenado los coeficientes δ_i, donde ahora se satisface la siguiente desigualdad $\widehat{\delta}_i \leq \widehat{\delta}_{i+1}$. Entonces la sumatoria adecuada para ser implementada en *double precision* considerando que se suma del índice menor hasta el mayor es la siguiente,

$$\tau_n(x) = \sum_{i=0}^{n-1} \widehat{\delta}_i.$$

(b) El código a continuación presenta las componentes principales a evaluar.

```
'''
input:
x       : (double) "x" value.
gammas  : (ndarray) gamma_i values in a vector of dimension "n".
n       : (integer) Associated to upper limit of sum.

output:
tau     : (double) Value of tau_n(x).
'''
def compute_tau_n(x,gammas,n):
```

- **[10 puntos]:** Construcción de vector $[1, x, x^2, \ldots, x^{n-1}]$ de forma vectorizada.
```
i = np.arange(n)
xp = np.power(x,i)
```
- **[5 puntos]:** Construcción de los coeficientes δ_i de forma vectorizada, es decir $[\gamma_0, \gamma_1\, x, \gamma_2\, x^2, \ldots, \gamma_{n-1}\, x^{n-1}]$.
```
delta = gammas * xp
```
- **[5 puntos]:** Ordenar de menor a mayor los coeficientes δ_i, es decir, construir $\widehat{\delta}_i$.
```
delta_hat = np.sort(delta)
```
- **[5 puntos]:** Sumar los coeficientes $\widehat{\delta}_i$ desde el menor, es decir el *primero*, hasta el mayor, es decir el *último*.
```
tau = 0.
for di in delta_hat:
    tau = tau + di
return tau
```