Questao 8

Codigo Fonte:

```
import io
from contextlib import redirect_stdout
from fpdf import FPDF
from fpdf.enums import XPos, YPos
import warnings
warnings.filterwarnings('ignore')

def solve_and_capture_output():
    """
    Executa a tradução das sentenças para Lógica de Primeira Ordem.
    """
    print("Questao 8: Representacoes Logicas para Modus Ponens Generalizado")
    print("=" * 75)
    print("O objetivo e converter as sentencas em um formato de implicacao (regra),")
    print("como 'Para todo x, se P(x) entao Q(x)', sempre que possivel.")
    print("-" * 75)

    # --- Sentença a) ---
    print("\na) Cavalos, vacas e porcos sao mamiferos.")
    print("   Analise:")
    print("   - Esta sentenca afirma que a propriedade de ser cavalo, vaca ou porco")
    print("     implica a propriedade de ser mamifero. Podemos quebra-la em tres regras.")
    print("   - Predicados: Cavalo(x), Vaca(x), Porco(x), Mamifero(x).")
    print("\n   Traducao Logica:")
    print("   1. Ax, Cavalo(x) -> Mamifero(x)")
    print("   2. Ax, Vaca(x) -> Mamifero(x)")
    print("   3. Ax, Porco(x) -> Mamifero(x)")
    print("-" * 75)

    # --- Sentença b) ---
    print("\nb) A prole de um cavalo e um cavalo.")
    print("   Analise:")
    print("   - Significa que 'Para todo x e y, se y e a prole de x E x e um cavalo,")
    print("     entao y tambem e um cavalo'.")
    print("   - Predicados: Prole(y, x) [y e a prole de x], Cavalo(x).")
    print("\n   Traducao Logica:")
    print("   Ax,y, (Prole(y, x) ^ Cavalo(x)) -> Cavalo(y)")
    print("-" * 75)

    # --- Sentença c) ---
    print("\nc) Barba Azul e um cavalo.")
    print("   Analise:")
    print("   - Esta e uma sentenca atomica, um fato. Nao e uma regra (implicacao).")
    print("     Ela e usada como uma premissa para iniciar inferencias.")
    print("   - Constante: BarbaAzul.")
    print("   - Predicado: Cavalo(x).")
    print("\n   Traducao Logica:")
```

```
    print("   Cavalo(BarbaAzul)")
    print("-" * 75)

    # --- Sentença d) ---
    print("\nd) Barba Azul e o pai ou mae de Charlie.")
    print("   Analise:")
    print("   - Outro fato, que estabelece uma relacao entre duas constantes.")
    print("   - Constantes: BarbaAzul, Charlie.")
    print("   - Predicado: PaiOuMae(p, a) [p e pai ou mae de a].")
    print("\n   Traducao Logica:")
    print("   PaiOuMae(BarbaAzul, Charlie)")
    print("-" * 75)

    # --- Sentença e) ---
    print("\ne) Prole e pai ou mae sao relacoes inversas.")
    print("   Analise:")
    print("   - Isso define a relacao entre os predicados. Uma relacao inversa")
    print("     e uma bi-implicacao (se e somente se), que podemos representar")
    print("     como duas regras de implicacao separadas.")
    print("   - Predicados: PaiOuMae(x, y), Prole(y, x).")
    print("\n   Traducao Logica:")
    print("   1. Ax,y, PaiOuMae(x, y) -> Prole(y, x)")
    print("   2. Ax,y, Prole(y, x) -> PaiOuMae(x, y)")
    print("-" * 75)

    # --- Sentença f) ---
    print("\nf) Todo mamifero tem um pai ou mae.")
    print("   Analise:")
    print("   - A traducao direta e 'Para todo x, se x e um mamifero, entao existe")
    print("     um y tal que y e pai ou mae de x'.")
    print("   - A presenca do 'existe' (E) no consequente de uma implicacao")
    print("     e tratada em sistemas de inferencia atraves de um processo chamado")
    print("     'Skolemizacao', que introduz uma funcao (ex: GenitorDe(x)) para")
    print("     representar esse 'y' que deve existir.")
    print("   - Predicados: Mamifero(x), PaiOuMae(y, x).")
    print("   - Funcao Skolem: GenitorDe(x).")
    print("\n   Traducao Logica Direta:")
    print("   Ax, Mamifero(x) -> (Ey, PaiOuMae(y, x))")
    print("\n   Traducao Logica Adequada para Inferencia (Skolemizada):")
    print("   Ax, Mamifero(x) -> PaiOuMae(GenitorDe(x), x)")
    print("=" * 75)


def generate_pdf_report(code_content, output_content):
    """Gera um PDF com o código e o output."""
    pdf = FPDF()
    pdf.add_page()
    pdf.set_font("Courier", 'B', 16)
    pdf.cell(0, 10, 'Questao 8', new_x=XPos.LMARGIN, new_y=YPos.NEXT, align='C')
    pdf.ln(10)

    pdf.set_font("Courier", 'B', 12)
    pdf.cell(0, 10, 'Codigo Fonte:', new_x=XPos.LMARGIN, new_y=YPos.NEXT)
```

```
    pdf.set_font("Courier", '', 8)
    pdf.multi_cell(0, 5, code_content)

    pdf.add_page()

    pdf.set_font("Courier", 'B', 12)
    pdf.cell(0, 10, 'Output da Execucao (Analise Logica):', new_x=XPos.LMARGIN, new_y=YPos.NEXT)
    pdf.set_font("Courier", '', 10)
    output_content_safe = output_content.encode('latin-1', 'replace').decode('latin-1')
    pdf.multi_cell(0, 5, output_content_safe)

    pdf_file_name = "resultado_questao_8.pdf"
    pdf.output(pdf_file_name)
    return pdf_file_name

# --- Bloco Principal de Execução ---
if __name__ == "__main__":
    output_buffer = io.StringIO()
    with redirect_stdout(output_buffer):
        solve_and_capture_output()
    output_content = output_buffer.getvalue()

    print("--- [INICIO] Resultado da Execucao no Terminal ---")
    print(output_content)
    print("--- [FIM] Resultado da Execucao no Terminal ---")

    try:
        with open(__file__, 'r', encoding='utf-8') as f:
            code_content = f.read()
    except Exception as e:
        code_content = f"Nao foi possivel ler o arquivo do codigo: {e}"

    try:
        pdf_file = generate_pdf_report(code_content, output_content)
        print(f"\nPDF '{pdf_file}' gerado com sucesso no diretorio atual!")
    except Exception as e:
        print(f"\nOcorreu um erro ao gerar o PDF: {e}")
```

Output da Execucao (Analise Logica):

```
Questao 8: Representacoes Logicas para Modus Ponens Generalizado
================================================================================
O objetivo e converter as sentencas em um formato de implicacao (regra),
como 'Para todo x, se P(x) entao Q(x)', sempre que possivel.
--------------------------------------------------------------------------------

a) Cavalos, vacas e porcos sao mamiferos.
   Analise:
   - Esta sentenca afirma que a propriedade de ser cavalo, vaca ou porco
     implica a propriedade de ser mamifero. Podemos quebra-la em tres regras.
   - Predicados: Cavalo(x), Vaca(x), Porco(x), Mamifero(x).

   Traducao Logica:
   1. Ax, Cavalo(x) -> Mamifero(x)
   2. Ax, Vaca(x) -> Mamifero(x)
   3. Ax, Porco(x) -> Mamifero(x)
--------------------------------------------------------------------------------

b) A prole de um cavalo e um cavalo.
   Analise:
   - Significa que 'Para todo x e y, se y e a prole de x E x e um cavalo,
     entao y tambem e um cavalo'.
   - Predicados: Prole(y, x) [y e a prole de x], Cavalo(x).

   Traducao Logica:
   Ax,y, (Prole(y, x) ^ Cavalo(x)) -> Cavalo(y)
--------------------------------------------------------------------------------

c) Barba Azul e um cavalo.
   Analise:
   - Esta e uma sentenca atomica, um fato. Nao e uma regra (implicacao).
     Ela e usada como uma premissa para iniciar inferencias.
   - Constante: BarbaAzul.
   - Predicado: Cavalo(x).

   Traducao Logica:
   Cavalo(BarbaAzul)
--------------------------------------------------------------------------------

d) Barba Azul e o pai ou mae de Charlie.
   Analise:
   - Outro fato, que estabelece uma relacao entre duas constantes.
   - Constantes: BarbaAzul, Charlie.
   - Predicado: PaiOuMae(p, a) [p e pai ou mae de a].

   Traducao Logica:
   PaiOuMae(BarbaAzul, Charlie)
--------------------------------------------------------------------------------

e) Prole e pai ou mae sao relacoes inversas.
   Analise:
```

- Isso define a relacao entre os predicados. Uma relacao inversa
  e uma bi-implicacao (se e somente se), que podemos representar
  como duas regras de implicacao separadas.
- Predicados: PaiOuMae(x, y), Prole(y, x).

Traducao Logica:
1. Ax,y, PaiOuMae(x, y) -> Prole(y, x)
2. Ax,y, Prole(y, x) -> PaiOuMae(x, y)

---

f) Todo mamifero tem um pai ou mae.
Analise:
- A traducao direta e 'Para todo x, se x e um mamifero, entao existe
  um y tal que y e pai ou mae de x'.
- A presenca do 'existe' (E) no consequente de uma implicacao
  e tratada em sistemas de inferencia atraves de um processo chamado
  'Skolemizacao', que introduz uma funcao (ex: GenitorDe(x)) para
  representar esse 'y' que deve existir.
- Predicados: Mamifero(x), PaiOuMae(y, x).
- Funcao Skolem: GenitorDe(x).

Traducao Logica Direta:
Ax, Mamifero(x) -> (Ey, PaiOuMae(y, x))

Traducao Logica Adequada para Inferencia (Skolemizada):
Ax, Mamifero(x) -> PaiOuMae(GenitorDe(x), x)

==========