Program 9:Create a knowledge base consisting of first order logic statements and prove the given query using Resolution

```
# Define the knowledge base (KB)
KB = {
    "food(Apple)": True,
    "food(vegetables)": True,
    "eats(Anil, Peanuts)": True,
    "alive(Anil)": True,
    "likes(John, X)": "food(X)", # Rule: John likes all food
    "food(X)": "eats(Y, X) and not killed(Y)", # Rule: Anything eaten and not killed is food
    "eats(Harry, X)": "eats(Anil, X)", # Rule: Harry eats what Anil eats
    "alive(X)": "not killed(X)", # Rule: Alive implies not killed
    "not killed(X)": "alive(X)", # Rule: Not killed implies alive
}

# Function to evaluate if a predicate is true based on the KB
def resolve(predicate):
    # If it's a direct fact in KB
    if predicate in KB and isinstance(KB[predicate], bool):
        return KB[predicate]

    # If it's a derived rule
    if predicate in KB:
        rule = KB[predicate]
```

```
    if " and " in rule: # Handle conjunction
        sub_preds = rule.split(" and ")
        return all(resolve(sub.strip()) for sub in sub_preds)
    elif " or " in rule: # Handle disjunction
        sub_preds = rule.split(" or ")
        return any(resolve(sub.strip()) for sub in sub_preds)
    elif "not " in rule: # Handle negation
        sub_pred = rule[4:]  # Remove "not "
        return not resolve(sub_pred.strip())
    else: # Handle single predicate
        return resolve(rule.strip())

# If the predicate is a specific query (e.g., likes(John, Peanuts))
if "(" in predicate:
    func, args = predicate.split("(")
    args = args.strip(")").split(", ")
    if func == "food" and args[0] == "Peanuts":
        return resolve("eats(Anil, Peanuts)") and not resolve("killed(Anil)")
    if func == "likes" and args[0] == "John" and args[1] == "Peanuts":
        return resolve("food(Peanuts)")

# Default to False if no rule or fact applies
return False
```

```
# Query to prove: John likes Peanuts

query = "likes(John, Peanuts)"

result = resolve(query)


# Print the result

print(f"Does John like peanuts? {'Yes' if result else 'No'}")
```

```
Does John like peanuts? Yes
```