In [1]:
```
# Extractive question answering using Transformers developed by Hugging Face,
 Inc.
# Note: Transformers were installed with 'python -m pip install transformers'
 in pytorch environment (pytorchenv active)
```

In [2]:
```
# import necessary transformers classes and torch

from transformers import AutoTokenizer, AutoModelForQuestionAnswering
import torch
```

In [3]:
```
# instantiate pretrained tokenizer and model

tokenizer = AutoTokenizer.from_pretrained("bert-large-uncased-whole-word-maski
ng-finetuned-squad")
model = AutoModelForQuestionAnswering.from_pretrained("bert-large-uncased-whol
e-word-masking-finetuned-squad")
```

In [4]:
```
# provide text and questions

# text from Wikipedea: https://en.wikipedia.org/wiki/Transformer_(machine_lear
ning_model)
text = r"""
The Transformer is a deep machine learning model introduced in 2017, used prim
arily in the field of natural language processing
(NLP).[1] Like recurrent neural networks (RNNs), Transformers are designed to
 handle ordered sequences of data,
such as natural language, for various tasks such as machine translation and te
xt summarization.
However, unlike RNNs, Transformers do not require that the sequence be process
ed in order. So, if the data in question is
natural language, the Transformer does not need to process the beginning of a
 sentence before it processes the end.
Due to this feature, the Transformer allows for much more parallelization than
RNNs during training.[1]

Since their introduction, Transformers have become the basic building block of
most state-of-the-art architectures in NLP,
replacing gated recurrent neural network models such as the long short-term me
mory (LSTM) in many cases.
Since the Transformer architecture facilitates more parallelization during tra
ining computations,
it has enabled training on much more data than was possible before it was intr
oduced.
This led to the development of pretrained systems such as BERT (Bidirectional
 Encoder Representations from Transformers)
and GPT-2, which have been trained with huge amounts of general language data
 prior to being released,
and can then be fine-tune trained to specific language tasks.
"""

questions = [
    "What is Transformer?",
    "Whent was Transformer introduced?",
    "What are Transformers designed for?",
    "What tasks can Transformer handle?",
    "What don't Transformers require?",
    "What have Transformers become?",
    "What does Transformer architecture facilitate?",
    "What did Transformer lead to?"
]
```

In [5]:
```
# answer questions

for question in questions:
    inputs = tokenizer.encode_plus(question, text, add_special_tokens=True, re
turn_tensors="pt")
    input_ids = inputs["input_ids"].tolist()[0]

    text_tokens = tokenizer.convert_ids_to_tokens(input_ids)
    answer_start_scores, answer_end_scores = model(**inputs)

    answer_start = torch.argmax(answer_start_scores)  # get the most likely be
ginning of answer with the argmax of the score
    answer_end = torch.argmax(answer_end_scores) + 1  # get the most likely en
d of answer with the argmax of the score

    answer = tokenizer.convert_tokens_to_string(tokenizer.convert_ids_to_token
s(input_ids[answer_start:answer_end]))

    print(f"Question: {question}")
    print(f"Answer: {answer}\n")
```

```
Question: What is Transformer?
Answer: a deep machine learning model

Question: Whent was Transformer introduced?
Answer: 2017

Question: What are Transformers designed for?
Answer: to handle ordered sequences of data

Question: What tasks can Transformer handle?
Answer: machine translation and text summarization

Question: What don't Transformers require?
Answer: the sequence be processed in order

Question: What have Transformers become?
Answer: the basic building block of most state - of - the - art architectures

Question: What does Transformer architecture facilitate?
Answer: more parallelization during training computations

Question: What did Transformer lead to?
Answer: development of pretrained systems
```

In [6]:
```
# All questions have been answered!
```