Fibonacci sequence calculation in python?

Python function that calculates fibonacci sequence?

```python
def fib(n):
    a, b = 0, 1
    while a < n:
        print(a, end=" ")
        a, b = b, a + b
    print()
```

Add a comment

1 Answer

Sorted by: Highest score (default)
def max(a, b):
    """Return the maximum of two numbers"""
    if a > b:
        return a
    else:
        return b
Natural Language vs. Code

• Very different structure
• Code has underlying logic, which strongly differs from NL

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Natural Language</th>
<th>Programming Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>Syntax</td>
<td>Unstructured and often ambiguous</td>
<td>Precise and structured</td>
</tr>
<tr>
<td>Vocabulary</td>
<td>Vast and diverse</td>
<td>Limited and well-defined</td>
</tr>
<tr>
<td>Ambiguity</td>
<td>Common</td>
<td>Rare</td>
</tr>
<tr>
<td>Error Tolerance</td>
<td>Tolerant of errors</td>
<td>Strict</td>
</tr>
</tbody>
</table>
Syntax/Ambiguity

The burglar threatened the student with the knife

```
1  def myFunction(myArguments):
2      # do something
3      myResult = process(myArguments)
4      return myResult
```
Vocabulary

- Over 150k English words
- Python has 33 reserved keywords

<table>
<thead>
<tr>
<th>Python Keywords</th>
</tr>
</thead>
<tbody>
<tr>
<td>False</td>
</tr>
<tr>
<td>None</td>
</tr>
<tr>
<td>True</td>
</tr>
<tr>
<td>and</td>
</tr>
<tr>
<td>as</td>
</tr>
<tr>
<td>assert</td>
</tr>
<tr>
<td>break</td>
</tr>
<tr>
<td>class</td>
</tr>
<tr>
<td>continue</td>
</tr>
</tbody>
</table>

RealPython: https://realpython.com/lessons/reserved-keywords/
Error Tolerance

Yuo cna porbalby raed tihs esaliy desptie teh msispeillgns

```python
# function with error
def max(a, b):
    if a > b:
        return a
    else:
        return b
```
Natural Language vs. Code

• Very different structure
• Code has underlying logic, which strongly differs from NL

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Natural Language</th>
<th>Programming Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>Syntax</td>
<td>Unstructured and often ambiguous</td>
<td>Precise and structured</td>
</tr>
<tr>
<td>Vocabulary</td>
<td>Vast and diverse</td>
<td>Limited and well-defined</td>
</tr>
<tr>
<td>Ambiguity</td>
<td>Common</td>
<td>Rare</td>
</tr>
<tr>
<td>Error Tolerance</td>
<td>Tolerant of errors</td>
<td>Strict</td>
</tr>
</tbody>
</table>
Abstract Syntax Tree

```
def max(b, a):
    x = 0
    if b > a:
        x = b
    else:
        x = a
    return x
```
Data Flow

1. ```
def max(b, a):
    x = 0
    if b > a:
        x = b
    else:
        x = a
    return x
```
Data Flow Derivation

Source code:
```python
def max(a, b):
    x = 0
    if b > a:
        x = b
    else:
        x = a
    return x
```

Parse into AST:

Identify variable sequence:
```python
def max(a, b):
    x = 0
    if b > a:
        x = b
    else:
        x = a
    return x
```

Variable relation:

Compiler Tool: Identify variable sequence in AST
Extract variable relation from AST

Value comes from
GraphCodeBERT architecture

**Source code**
```
def max(a, b):
    x = 0
    if b > a:
        x = b
    else:
        x = a
    return x
```

**Comment**
*Return maximum value*

**Data Flow**

Value comes from

GraphCodeBERT
Attention Mask

Ketan Doshi: https://towardsdatascience.com/transformers-explained-visually-part-2-how-it-works-step-by-step-b49fa4a64f34
PreTraining Tasks

• Masked Language Modeling
• Edge Prediction
• Node Alignment
Masked Language Modelling

• Common way of pretraining
• Follows BERT approach

```python
1 def max(a, b):
2     """Return maximum value"""
3     x = 0
4     if b > a:
5         x = [MASK]
6     else:
7         x = a
8     return x
```

```python
1 def max(a, b):
2     """Return [MASK] value"""
3     x = 0
4     if b > a:
5         x = b
6     else:
7         x = a
8     return x
```
Edge Prediction Pretraining

Source code

def max(a, b):
    if b > a:
        x = b
    else:
        x = a
    return x

Comment

Return maximum value

Data Flow

\[
\text{loss}_{\text{Edge Pred}} = - \sum_{e_{ij} \in E_c} [\delta(e_{ij} \in E_{\text{mask}}) \log p_{e_{ij}} + (1 - \delta(e_{ij} \in E_{\text{mask}})) \log(1 - p_{e_{ij}})]
\]
Node Alignment Pretraining

GraphCodeBERT

\[
\text{loss}_{\text{NodeAlign}} = -\sum_{e_{ij} \in E'_c} [\delta(e_{ij} \in E'_\text{mask}) \log p_{e_{ij}} + (1 - \delta(e_{ij} \in E'_\text{mask})) \log(1 - p_{e_{ij}})]
\]
RoBERTa

- Train longer, bigger batches and more data
- Remove NSP
- Train on longer sequences
- Dynamically change masking pattern

Scaler Topics: https://www.scaler.com/topics/nlp/bert-next-sentence-prediction/
CodeBERT

Source code

```python
def max(a, b):
    if b > a:
        x = b
    else:
        x = a
    return x
```

Comment

Return maximum value

Data Flow

Value comes from \( x^{11} \)

GraphCodeBERT
SynCoBERT

(a) SynCoBERT pre-training over MMLM, IP and TEP objectives

SynCoBERT paper: https://arxiv.org/abs/2108.04556
MRR Score

$$\text{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\text{rank}_i}.$$
## NL Code Search Results

<table>
<thead>
<tr>
<th>model</th>
<th>Ruby</th>
<th>Javascript</th>
<th>Go</th>
<th>Python</th>
<th>Java</th>
<th>Php</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>NBow</td>
<td>0.162</td>
<td>0.157</td>
<td>0.330</td>
<td>0.161</td>
<td>0.171</td>
<td>0.152</td>
<td>0.189</td>
</tr>
<tr>
<td>CNN</td>
<td>0.276</td>
<td>0.224</td>
<td>0.680</td>
<td>0.242</td>
<td>0.263</td>
<td>0.260</td>
<td>0.324</td>
</tr>
<tr>
<td>BiRNN</td>
<td>0.213</td>
<td>0.193</td>
<td>0.688</td>
<td>0.290</td>
<td>0.304</td>
<td>0.338</td>
<td>0.338</td>
</tr>
<tr>
<td>selfAtt</td>
<td>0.275</td>
<td>0.287</td>
<td>0.723</td>
<td>0.398</td>
<td>0.404</td>
<td>0.426</td>
<td>0.419</td>
</tr>
<tr>
<td>RoBERTa</td>
<td>0.587</td>
<td>0.517</td>
<td>0.850</td>
<td>0.587</td>
<td>0.599</td>
<td>0.560</td>
<td>0.617</td>
</tr>
<tr>
<td>RoBERTa (code)</td>
<td>0.628</td>
<td>0.562</td>
<td>0.859</td>
<td>0.610</td>
<td>0.620</td>
<td>0.579</td>
<td>0.643</td>
</tr>
<tr>
<td>CodeBERT</td>
<td>0.679</td>
<td>0.620</td>
<td>0.882</td>
<td>0.672</td>
<td>0.676</td>
<td>0.628</td>
<td>0.693</td>
</tr>
<tr>
<td>GraphCodeBERT</td>
<td><strong>0.703</strong></td>
<td><strong>0.644</strong></td>
<td><strong>0.897</strong></td>
<td><strong>0.692</strong></td>
<td><strong>0.691</strong></td>
<td><strong>0.649</strong></td>
<td><strong>0.713</strong></td>
</tr>
</tbody>
</table>
## SynCoBERT NL Code Search Results

<table>
<thead>
<tr>
<th>Model</th>
<th>AdvTest</th>
<th>CodeSearch</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Python</td>
<td>Ruby</td>
</tr>
<tr>
<td>NBow</td>
<td>-</td>
<td>16.2</td>
</tr>
<tr>
<td>CNN</td>
<td>-</td>
<td>27.6</td>
</tr>
<tr>
<td>BiRNN</td>
<td>-</td>
<td>21.3</td>
</tr>
<tr>
<td>Transformer</td>
<td>-</td>
<td>27.5</td>
</tr>
<tr>
<td>RoBERTa</td>
<td>18.3</td>
<td>58.7</td>
</tr>
<tr>
<td>RoBERTa (code)</td>
<td>-</td>
<td>62.8</td>
</tr>
<tr>
<td>CodeBERT</td>
<td>27.2</td>
<td>67.9</td>
</tr>
<tr>
<td>GraphCodeBERT</td>
<td>35.2</td>
<td>70.3</td>
</tr>
<tr>
<td>SYNCObERT</td>
<td><strong>38.1</strong></td>
<td><strong>72.2</strong></td>
</tr>
</tbody>
</table>
## Ablation Study

<table>
<thead>
<tr>
<th>Methods</th>
<th>Ruby</th>
<th>Javascript</th>
<th>Go</th>
<th>Python</th>
<th>Java</th>
<th>Php</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>GraphCodeBERT</td>
<td>0.703</td>
<td>0.644</td>
<td>0.897</td>
<td>0.692</td>
<td>0.691</td>
<td>0.649</td>
<td>0.713</td>
</tr>
<tr>
<td>-w/o EdgePred</td>
<td>0.701</td>
<td>0.632</td>
<td>0.894</td>
<td>0.687</td>
<td>0.688</td>
<td>0.640</td>
<td>0.707</td>
</tr>
<tr>
<td>-w/o NodeAlign</td>
<td>0.685</td>
<td>0.635</td>
<td>0.887</td>
<td>0.682</td>
<td>0.690</td>
<td>0.640</td>
<td>0.703</td>
</tr>
<tr>
<td>-w/o Data Flow</td>
<td>0.679</td>
<td>0.620</td>
<td>0.882</td>
<td>0.672</td>
<td>0.676</td>
<td>0.628</td>
<td>0.693</td>
</tr>
</tbody>
</table>
Code Clone Detection

• Measure similarity between two code segments
• Code segments have similar output for the same input
• Easier software maintenance and to prevent bugs
• BigCloneBench dataset
```java
protected String downloadURLtoString(URL url) throws IOException {
    BufferedReader in = new BufferedReader(new InputStreamReader(url.openStream()));
    StringBuffer sb = new StringBuffer(100 * 1024);
    String str;
    while ((str = in.readLine()) != null) {
        sb.append(str);
    }
    in.close();
    return sb.toString();
}
```

```java
public static String fetchUrl(String urlString) {
    try {
        URL url = new URL(urlString);
        BufferedReader reader = new BufferedReader(new InputStreamReader(url.openStream()));
        String line = null;
        StringBuilder builder = new StringBuilder();
        while (line = reader.readLine() != null) {
            builder.append(line);
        }
        reader.close();
        return builder.toString();
    } catch (MalformedURLException e) {
    } catch (IOException e) {
    } catch (Exception e) {
        return "";
    }
```
Clone Detection Results

<table>
<thead>
<tr>
<th>Model</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deckard</td>
<td>0.93</td>
<td>0.02</td>
<td>0.03</td>
</tr>
<tr>
<td>RtvNN</td>
<td>0.95</td>
<td>0.01</td>
<td>0.01</td>
</tr>
<tr>
<td>CDLH</td>
<td>0.92</td>
<td>0.74</td>
<td>0.82</td>
</tr>
<tr>
<td>ASTNN</td>
<td>0.92</td>
<td>0.94</td>
<td>0.93</td>
</tr>
<tr>
<td>FA-AST-GMN</td>
<td><strong>0.96</strong></td>
<td>0.94</td>
<td>0.95</td>
</tr>
<tr>
<td>RoBERTa (code)</td>
<td>0.949</td>
<td>0.922</td>
<td>0.935</td>
</tr>
<tr>
<td>CodeBERT</td>
<td>0.947</td>
<td>0.934</td>
<td>0.941</td>
</tr>
<tr>
<td>GraphCodeBERT</td>
<td>0.948</td>
<td><strong>0.952</strong></td>
<td><strong>0.950</strong></td>
</tr>
</tbody>
</table>
Code Translation

• Migrate code to different language
• Mostly used for legacy software
• BLEU score
• Dataset crawled from open source projects
BLEU Score

• Bilingual evaluation understudy
• Measure for similarity of machine translated text
• Based on frequency of shared words and phrases
• Frequencies are compared with reference corpus

The closer a machine translation is to a professional human translation, the better it is

BLEU Paper: https://aclanthology.org/P02-1040.pdf
Java

```java
1 public void print(boolean b){
2     print(String.valueOf(b));
3 }
```

C#

```csharp
1 public void print(bool b){
2     print(b.ToString());
3 }
```
## Code Translation Results

<table>
<thead>
<tr>
<th>Method</th>
<th>Java→C#</th>
<th>C#→Java</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BLEU</td>
<td>Acc</td>
</tr>
<tr>
<td>Naive</td>
<td>18.54</td>
<td>0.0</td>
</tr>
<tr>
<td>PBSMT</td>
<td>43.53</td>
<td>12.5</td>
</tr>
<tr>
<td>Transformer</td>
<td>55.84</td>
<td>33.0</td>
</tr>
<tr>
<td>RoBERTa (code)</td>
<td>77.46</td>
<td>56.1</td>
</tr>
<tr>
<td>CodeBERT</td>
<td>79.92</td>
<td>59.0</td>
</tr>
<tr>
<td>GraphCodeBERT</td>
<td><strong>80.58</strong></td>
<td><strong>59.4</strong></td>
</tr>
</tbody>
</table>
Code Refinement

- Automatically fix bugs
- Reduces cost of bug fixes
- Dataset by Tufano et al.
## Code Refinement Results

<table>
<thead>
<tr>
<th>Method</th>
<th>small</th>
<th>medium</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BLEU</td>
<td>Acc</td>
</tr>
<tr>
<td>Naive</td>
<td>78.06</td>
<td>0.0</td>
</tr>
<tr>
<td>LSTM</td>
<td>76.76</td>
<td>10.0</td>
</tr>
<tr>
<td>Transformer</td>
<td>77.21</td>
<td>14.7</td>
</tr>
<tr>
<td>RoBERTa (code)</td>
<td>77.30</td>
<td>15.9</td>
</tr>
<tr>
<td>CodeBERT</td>
<td>77.42</td>
<td>16.4</td>
</tr>
<tr>
<td>GraphCodeBERT</td>
<td>80.02</td>
<td>17.3</td>
</tr>
</tbody>
</table>
Conclusion/Remarks

• Small improvement compared to CodeBERT
• SynCoBERT outperformed the model
• Not clear benefits of Data Flow when compared with AST
• BLEU score is not really justified