```
faqs = """About the Program
What is the course fee for  Data Science Mentorship Program (DSMP
2023)
The course follows a monthly subscription model where you have to make
monthly payments of Rs 799/month.
What is the total duration of the course?
The total duration of the course is 7 months. So the total course fee
becomes 799*7 = Rs 5600(approx.)
What is the syllabus of the mentorship program?
We will be covering the following modules:
Python Fundamentals
Python libraries for Data Science
Data Analysis
SQL for Data Science
Maths for Machine Learning
ML Algorithms
Practical ML
MLOPs
Will Deep Learning and NLP be a part of this program?
No, NLP and Deep Learning both are not a part of this program's
curriculum.
What if I miss a live session? Will I get a recording of the session?
Yes all our sessions are recorded, so even if you miss a session you
can go back and watch the recording.
What is the time duration of all the live sessions?
Roughly, all the sessions last 2 hours.
What is the language spoken by the instructor during the sessions?
Hinglish
How will I be informed about the upcoming class?
You will get a mail from our side before every paid session once you
become a paid user.
Can I do this course if I am from a non-tech background?
Yes, absolutely.
I am late, can I join the program in the middle?
Absolutely, you can join the program anytime.
If I join/pay in the middle, will I be able to see all the past
lectures?
Yes, once you make the payment you will be able to see all the past
content in your dashboard.
Where do I have to submit the task?
You don't have to submit the task. We will provide you with the
solutions, you have to self evaluate the task yourself.
Will we do case studies in the program?
Yes.
Payment/Registration related questions
Can we pay the entire amount of Rs 5600 all at once?
Unfortunately no, the program follows a monthly subscription model.
What is the validity of monthly subscription? Suppose if I pay on 15th
Jan, then do I have to pay again on 1st Feb or 15th Feb
```

```
15th Feb. The validity period is 30 days from the day you make the payment. So essentially you can join anytime you don’t have to wait for a month to end.
What if I don’t like the course after making the payment. What is the refund policy?
You get a 7 days refund period from the day you have made the payment.
Post registration queries
Till when can I view the paid videos on the website?
This one is tricky, so read carefully. You can watch the videos till your subscription is valid. Suppose you have purchased subscription on 21st Jan, you will be able to watch all the past paid sessions in the period of 21st Jan to 20th Feb. But after 21st Feb you will have to purchase the subscription again.
But once the course is over and you have paid us Rs 5600(or 7 installments of Rs 799) you will be able to watch the paid sessions till Aug 2024.
Why lifetime validity is not provided?
Because of the low course fee.
Where can I reach out in case of a doubt after the session?
You will have to fill a google form provided in your dashboard and our team will contact you for a 1 on 1 doubt clearance session
If I join the program late, can I still ask past week doubts?
Yes, just select past week doubt in the doubt clearance google form.
Certificate and Placement Assistance related queries
What is the criteria to get the certificate?
There are 2 criterias:
You have to pay the entire fee of Rs 5600
You have to attempt all the course assessments.
I am joining late. How can I pay payment of the earlier months?
You will get a link to pay fee of earlier months in your dashboard once you pay for the current month.
I have read that Placement assistance is a part of this program. What comes under Placement assistance?
This is to clarify that Placement assistance does not mean Placement guarantee. So we dont guarantee you any jobs or for that matter even interview calls. So if you are planning to join this course just for placements, I am afraid you will be disappointed. Here is what comes under placement assistance
Portfolio Building sessions
Soft skill sessions
Sessions with industry mentors
Discussion on Job hunting strategies
"""

import tensorflow as tf
from tensorflow.keras.preprocessing.text import Tokenizer

tokenizer = Tokenizer()

tokenizer.fit_on_texts([faqs])
```

```
len(tokenizer.word_index)
```

```
251
```

```
tokenizer.word_index["nlp"]
```

```
84
```

```
input_sequences = []
for sentence in faqs.split('\n'):
  tokenized_sentence = tokenizer.texts_to_sequences([sentence])[0]

  for i in range(1,len(tokenized_sentence)):
    input_sequences.append(tokenized_sentence[:i+1])
```

```
input_sequences[:20]
```

```
[[77, 1],
 [77, 1, 13],
 [10, 6],
 [10, 6, 1],
 [10, 6, 1, 11],
 [10, 6, 1, 11, 32],
 [10, 6, 1, 11, 32, 14],
 [10, 6, 1, 11, 32, 14, 40],
 [10, 6, 1, 11, 32, 14, 40, 52],
 [10, 6, 1, 11, 32, 14, 40, 52, 78],
 [10, 6, 1, 11, 32, 14, 40, 52, 78, 13],
 [10, 6, 1, 11, 32, 14, 40, 52, 78, 13, 123],
 [10, 6, 1, 11, 32, 14, 40, 52, 78, 13, 123, 124],
 [1, 11],
 [1, 11, 79],
 [1, 11, 79, 7],
 [1, 11, 79, 7, 41],
 [1, 11, 79, 7, 41, 22],
 [1, 11, 79, 7, 41, 22, 80],
 [1, 11, 79, 7, 41, 22, 80, 53]]
```

```
max_len = max([len(x) for x in input_sequences])
max_len
```

```
57
```

```
from tensorflow.keras.preprocessing.sequence import pad_sequences
padded_input_sequences = pad_sequences(input_sequences, maxlen =
max_len, padding='pre')
```

```
padded_input_sequences
```

```
array([[  0,   0,   0, ...,   0,  77,   1],
       [  0,   0,   0, ...,  77,   1,  13],
       [  0,   0,   0, ...,   0,  10,   6],
```

```
       ...,
       [  0,   0,   0, ..., 248,  30, 249],
       [  0,   0,   0, ...,  30, 249, 250],
       [  0,   0,   0, ..., 249, 250, 251]])
```

```
X = padded_input_sequences[:,:-1]
```

```
y = padded_input_sequences[:,-1]
```

```
X.shape
```

```
(704, 56)
```

```
y.shape
```

```
(704,)
```

```
from tensorflow.keras.utils import to_categorical
y = to_categorical(y,num_classes=len(tokenizer.word_index)+1)
```

```
y.shape
```

```
(704, 252)
```

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, LSTM, Dense

model = Sequential()
model.add(Embedding(len(tokenizer.word_index)+1, 100))
model.add(LSTM(150, return_sequences=True)) # Added
return_sequences=True
model.add(LSTM(150))
model.add(Dense(len(tokenizer.word_index)+1, activation='softmax'))

model.compile(loss='categorical_crossentropy',
optimizer='adam',metrics=['accuracy'])

model.fit(X,y,epochs=75)
```

```
Epoch 1/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 2s 29ms/step - accuracy: 0.0570 - loss:
5.4329
Epoch 2/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 29ms/step - accuracy: 0.0899 - loss:
4.9516
Epoch 3/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.0738 - loss:
4.8953
Epoch 4/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 29ms/step - accuracy: 0.1107 - loss:
4.8408
Epoch 5/75
```

```
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 29ms/step - accuracy: 0.0906 - loss:
4.8633
Epoch 6/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.0891 - loss:
4.8537
Epoch 7/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.0831 - loss:
4.8175
Epoch 8/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 29ms/step - accuracy: 0.0727 - loss:
4.8835
Epoch 9/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.1088 - loss:
4.6207
Epoch 10/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 29ms/step - accuracy: 0.0973 - loss:
4.6764
Epoch 11/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.0937 - loss:
4.6482
Epoch 12/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.0811 - loss:
4.5960
Epoch 13/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 32ms/step - accuracy: 0.1051 - loss:
4.4714
Epoch 14/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 32ms/step - accuracy: 0.1005 - loss:
4.4375
Epoch 15/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.0888 - loss:
4.3234
Epoch 16/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 32ms/step - accuracy: 0.1175 - loss:
4.1676
Epoch 17/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 32ms/step - accuracy: 0.1348 - loss:
4.0137
Epoch 18/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.1521 - loss:
3.8872
Epoch 19/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.1636 - loss:
3.7948
Epoch 20/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.1925 - loss:
3.6393
Epoch 21/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.1818 - loss:
```

```
3.5055
Epoch 22/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.2220 - loss:
3.3961
Epoch 23/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 32ms/step - accuracy: 0.2552 - loss:
3.2026
Epoch 24/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 32ms/step - accuracy: 0.2901 - loss:
3.1204
Epoch 25/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.3150 - loss:
2.9875
Epoch 26/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.3488 - loss:
2.8978
Epoch 27/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 32ms/step - accuracy: 0.3469 - loss:
2.8076
Epoch 28/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.3807 - loss:
2.6228
Epoch 29/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.4347 - loss:
2.5166
Epoch 30/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.4638 - loss:
2.3909
Epoch 31/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.4984 - loss:
2.2781
Epoch 32/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.5125 - loss:
2.2011
Epoch 33/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.5202 - loss:
2.1745
Epoch 34/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 29ms/step - accuracy: 0.5583 - loss:
1.9982
Epoch 35/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.6019 - loss:
1.9108
Epoch 36/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.5986 - loss:
1.8796
Epoch 37/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 32ms/step - accuracy: 0.6598 - loss:
1.6833
```

```
Epoch 38/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.6582 - loss:
1.6649
Epoch 39/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.7091 - loss:
1.6097
Epoch 40/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.7106 - loss:
1.5492
Epoch 41/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.7271 - loss:
1.4226
Epoch 42/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.7097 - loss:
1.4146
Epoch 43/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.7753 - loss:
1.2759
Epoch 44/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.8041 - loss:
1.2072
Epoch 45/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.7578 - loss:
1.2038
Epoch 46/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.8180 - loss:
1.1046
Epoch 47/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.8307 - loss:
1.0893
Epoch 48/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.8597 - loss:
0.9859
Epoch 49/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.8441 - loss:
0.9997
Epoch 50/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.8822 - loss:
0.9152
Epoch 51/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.8857 - loss:
0.9000
Epoch 52/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.8673 - loss:
0.8817
Epoch 53/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.8849 - loss:
0.8296
Epoch 54/75
```

```
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9004 - loss:
0.7953
Epoch 55/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9136 - loss:
0.7652
Epoch 56/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9172 - loss:
0.6975
Epoch 57/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9125 - loss:
0.6964
Epoch 58/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9337 - loss:
0.6206
Epoch 59/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9258 - loss:
0.6081
Epoch 60/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9210 - loss:
0.5870
Epoch 61/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9302 - loss:
0.5795
Epoch 62/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9188 - loss:
0.5577
Epoch 63/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9458 - loss:
0.4915
Epoch 64/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9309 - loss:
0.5446
Epoch 65/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9355 - loss:
0.4865
Epoch 66/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9159 - loss:
0.4847
Epoch 67/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9422 - loss:
0.4445
Epoch 68/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 32ms/step - accuracy: 0.9413 - loss:
0.4037
Epoch 69/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9491 - loss:
0.3883
Epoch 70/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 32ms/step - accuracy: 0.9440 - loss:
```

```
0.4170
Epoch 71/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 32ms/step - accuracy: 0.9380 - loss:
0.3986
Epoch 72/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 32ms/step - accuracy: 0.9435 - loss:
0.3714
Epoch 73/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9468 - loss:
0.3636
Epoch 74/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 29ms/step - accuracy: 0.9500 - loss:
0.3472
Epoch 75/75
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9407 - loss:
0.3575

<keras.src.callbacks.history.History at 0x22a3d260050>
```

```
import numpy as np
import time
text = "what is the"

for i in range(10):
  # tokenize
  token_text = tokenizer.texts_to_sequences([text])[0]
  # padding
  padded_token_text = pad_sequences([token_text], maxlen=max_len,
padding='pre')
  # predict
  pos = np.argmax(model.predict(padded_token_text))

  for word,index in tokenizer.word_index.items():
    if index == pos:
      text = text + " " + word
      print(text)
      time.sleep(2)
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 592ms/step
what is the validity
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 46ms/step
what is the validity of
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 51ms/step
what is the validity of monthly
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 43ms/step
what is the validity of monthly subscription
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 48ms/step
what is the validity of monthly subscription suppose
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 42ms/step
what is the validity of monthly subscription suppose if
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 40ms/step
what is the validity of monthly subscription suppose if i
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 48ms/step
what is the validity of monthly subscription suppose if i pay
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 50ms/step
what is the validity of monthly subscription suppose if i pay on
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 41ms/step
what is the validity of monthly subscription suppose if i pay on 15th

model.summary()

Model: "sequential"

┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━┓
┃ Layer (type)                         ┃ Output Shape                ┃       Param # ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━┩
│ embedding (Embedding)                │ (32, 56, 100)               │        25,200 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ lstm (LSTM)                          │ (32, 56, 150)               │       150,600 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ lstm_1 (LSTM)                        │ (32, 150)                   │       180,600 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ dense (Dense)                        │ (32, 252)                   │        38,052 │
└──────────────────────────────────────┴─────────────────────────────┴─────────────────┘

 Total params: 1,183,358 (4.51 MB)

 Trainable params: 394,452 (1.50 MB)

 Non-trainable params: 0 (0.00 B)

 Optimizer params: 788,906 (3.01 MB)
```