Q1. How can you create a Bokeh plot using Python code?

To create a Bokeh plot in Python, you generally follow these steps:

1. Import required modules from `bokeh`
2. Create a `figure`
3. Add glyphs (visual shapes like lines, circles, bars)
4. Display or save the plot

Example

```
from bokeh.plotting import figure, show

# Create a figure
p = figure(title="Simple Line Plot",
           x_axis_label='X',
           y_axis_label='Y')

# Add a line glyph
p.line([1, 2, 3, 4], [2, 5, 8, 2], line_width=2)

# Show the plot
show(p)
```

Q2. What are glyphs in Bokeh, and how can you add them to a Bokeh plot? Explain with an example.

What are glyphs?

Glyphs are the **basic visual elements** used to represent data in Bokeh. Examples include:

- `line`

- `circle`
- `square`
- `rect`
- `vbar`
- `patch`

Each glyph maps data values to visual properties like position, size, and color.

Example: Adding glyphs

```
from bokeh.plotting import figure, show

p = figure(title="Glyph Example")

# Circle glyph
p.circle([1, 2, 3], [4, 6, 5], size=10, color="blue",
legend_label="Circles")

# Line glyph
p.line([1, 2, 3], [4, 6, 5], color="red", legend_label="Line")

show(p)
```

Q3. How can you customize the appearance of a Bokeh plot, including the axes, title, and legend?

You can customize a Bokeh plot by modifying properties of the figure and its components.

Customization Options

- **Title**: text, alignment, font size
- **Axes**: labels, ticks, grid lines

- **Legend**: location, orientation, visibility

Example

```
from bokeh.plotting import figure, show

p = figure(title="Customized Plot")

p.line([1, 2, 3], [3, 7, 4], legend_label="Data", line_width=2)

# Title customization
p.title.align = "center"
p.title.text_font_size = "16pt"

# Axis customization
p.xaxis.axis_label = "X Values"
p.yaxis.axis_label = "Y Values"
p.xaxis.major_label_text_font_size = "12pt"

# Legend customization
p.legend.location = "top_left"
p.legend.label_text_font_size = "10pt"

show(p)
```

Q4. What is a Bokeh server, and how can you use it to create interactive plots that can be updated in real time?

What is the Bokeh server?

The **Bokeh server** allows you to:

- Create **real-time, interactive visualizations**
- Keep Python code running on the server
- Update plots dynamically without reloading the page

It enables two-way communication between the browser and Python.

Example: Real-time updating plot

```
from bokeh.plotting import figure
from bokeh.models import ColumnDataSource
from bokeh.server.server import Server
import random

source = ColumnDataSource(data=dict(x=[], y=[]))

def modify_doc(doc):
    p = figure(title="Real-Time Data")
    p.line('x', 'y', source=source)

    def update():
        new_data = dict(
            x=[len(source.data['x'])],
            y=[random.random()]
        )
        source.stream(new_data, rollover=20)

    doc.add_periodic_callback(update, 1000)
    doc.add_root(p)

server = Server({'/': modify_doc})
server.start()
```

Run this script and open `http://localhost:5006` to see live updates.

Q5. How can you embed a Bokeh plot into a web page or dashboard using Flask or Django?

You can embed Bokeh plots using:

- `components()` for standalone embedding

- Bokeh server for advanced interactivity

Embedding in Flask

Flask App

```
from flask import Flask, render_template
from bokeh.plotting import figure
from bokeh.embed import components

app = Flask(__name__)

@app.route('/')
def index():
    p = figure(title="Bokeh in Flask")
    p.line([1, 2, 3], [4, 6, 2])

    script, div = components(p)
    return render_template("index.html", script=script, div=div)

app.run(debug=True)
```

HTML Template (`index.html`)

```
<!DOCTYPE html>
<html>
<head>
    {{ script | safe }}
</head>
<body>
    <h1>Bokeh Plot</h1>
    {{ div | safe }}
</body>
</html>
```

Embedding in Django (Conceptual Steps)

1. Generate Bokeh plot and extract `script` and `div`
2. Pass them to the Django view context
3. Render them safely in the template

```
from bokeh.embed import components
```

```
{{ script|safe }}
{{ div|safe }}
```
