Flask jsonify() - Cheat Sheet

```
1. What is jsonify?
-------------------
- Converts Python dict/list/tuple to JSON
- Returns Flask Response object
- Sets Content-Type to application/json

2. Return dict as JSON
----------------------
@app.route('/user')
def get_user():
    return jsonify({"id": 1, "name": "Siva"})

3. Return list as JSON
----------------------
@app.route('/users')
def get_users():
    users = [{"id": 1, "name": "Siva"}, {"id": 2, "name": "SK"}]
    return jsonify(users)

4. Multiple key-value pairs
---------------------------
@app.route('/status')
def get_status():
    return jsonify(status="success", code=200)

5. With HTTP status code
------------------------
return jsonify({"error": "Not Found"}), 404

6. Pretty print (debugging)
---------------------------
import json
print(json.dumps({"key": "value"}, indent=4))

7. Manual JSON response
-----------------------
from flask import Response
import json

@app.route('/manual')
def manual_json():
    data = {"msg": "hello"}
    return Response(json.dumps(data), mimetype='application/json')

8. jsonify() vs json.dumps()
----------------------------
jsonify():
- Returns Flask Response
- Auto sets content-type

json.dumps():
```

```
- Returns JSON string
- Needs manual Response wrapping

9. Serialization errors
------------------------
jsonify(set([1, 2, 3]))         # Error: set is not serializable
jsonify(datetime.now())        # Error: datetime is not serializable

10. Custom JSON Encoder
------------------------
import datetime

class CustomJSONEncoder(app.json_encoder):
    def default(self, obj):
        if isinstance(obj, datetime.datetime):
            return obj.isoformat()
        return super().default(obj)

app.json_encoder = CustomJSONEncoder
```