Codefest Challenges -5

Challenges -16:

CUDA-accelerated NN

Code:

```
import torch
import torch.nn as nn


class SimpleNN(nn.Module):
  def __init__(self):
    super(SimpleNN, self).__init__()
    self.model = nn.Sequential(
      nn.Linear(4, 5),
      nn.ReLU(),
      nn.Linear(5, 1)
    )

  def forward(self, x):
    return self.model(x)


model = SimpleNN().cuda()
x = torch.randn(1024, 4).cuda()
output = model(x)
```

Benchmark and Compare:

```
import time
```

```
start = time.time()
output = model(x)
torch.cuda.synchronize()
end = time.time()
print("PyTorch time:", end - start)
```

Challenge -17:

Code:

```
def systolic_bubble_sort(arr):
  n = len(arr)
  A = arr.copy()
  for i in range(n):
    for j in range(1 - i % 2, n - 1, 2):
      if A[j] > A[j + 1]:
        A[j], A[j + 1] = A[j + 1], A[j]
  return A

# Test
import random
arr = [random.randint(0, 100) for _ in range(100)]
print(systolic_bubble_sort(arr))
```

Benchmark + Compare:

```
import time
```

```
import matplotlib.pyplot as plt

sizes = [10, 100, 1000, 10000]
times = []

for size in sizes:
    arr = [random.randint(0, 10000) for _ in range(size)]
    start = time.time()
    systolic_bubble_sort(arr)
    times.append(time.time() - start)

plt.plot(sizes, times)
plt.xlabel('Array Size')
plt.ylabel('Time (s)')
plt.title('Systolic Bubble Sort Performance')
plt.grid(True)
plt.show()
```