Quant Guild Member Application

General Instructions

i. Keep the answers crisp and clear, remember to prefer quality over quantity.
ii. Write clean code with useful variable names and add comments to enhance readability.
iii. Feel free to make extensive use of ChatGPT. We are trying to test the logic behind your strategies, not your coding skills.
iv. Deadline to submit the application is 11:59 pm on 10th June 2025. A google form will be circulated 2 days before the deadline to submit your application.

We encourage you to put in your queries on the Whatsapp group, as it would help the people with the same doubt. If you still feel the necessity to, you can DM any of us, we will try to respond as soon as possible:
i) Ajoy Mathew - 7510710521
ii) Shashwat Kedia - 9331072210
iii) Lakshman Kishore - 9444980394

Quantifying You

Put in as many numbers as you can while answering the below questions. ~~PS: We love crunching on numbers!~~

1) Give a brief introduction about yourself.
2) What motivates you to join the Quant Guild?
3) Mention other commitments you have (or expect to have), for the next 2 semesters and how would you manage them along with Quant Guild.
4) Mention any past experiences (competitions, projects, etc) you have had in the Quant domain. (This question will not affect your application, it's just for us to know our audience better)

Problem Statement – Let’s Bid

Overview:

In this competition, each participant will program a trading bot that competes against other bots in a multi-round bidding game. The objective is to develop the most effective bidding strategy to maximize profit over several rounds.

Gameplay:

1) In each round, every player is given a value x_i (All x_i's are independent of each other i.e. they need not be the same for all players).
2) Each x_i is drawn from a distribution (specified in the next part).
3) Every player will submit a bid. Bids can be fractional as well, but must lie in the range [0, 100].
4) The player with the highest bid wins the item for that round. If multiple players are tied for the highest bid, all of them win the item.
5) Every player receives a payoff (explained in the section below)
6) Players start off with a fixed amount of capital, which gets updated according to their payoffs i.e. new capital = old capital + payoff.
7) At the start of each round, players will be provided with the following information:
   i. The highest and second highest bids of the previous 100 rounds
   ii. The amount of capital they have left
   iii. The number of players participating in that round of the auction.
8) Once a bot runs out of capital, it will no longer be able to participate in the future rounds in the auction.
9) If a player makes an illegal bid (bid > capital available, or bid does not lie in [0, min(100, capital available)]), then that player’s bid for the round will automatically be set to 0.

Auction Variants:

Variant 1 – Clock is Ticking

i. The value x_i for each player is drawn from a uniform random distribution over the range [0, 100]. All players know that the values are uniformly distributed.

ii. Every round, a clock will tick from 100 to 0. The bots need to return a time (bid) at which they would like to stop the clock. The first 2 bots to stop the clock (highest bids) win that round of the auction.

iii. The payoff for the winners is given by **Payoff = X – bid_i**; where bid_i is the bid of the ith bot and X is the max value amongst all x_i for that round.

iv. All other players receive **Payoff = 0.**

Variant 2 – Confidence is All you need

i. The value x_i for each player is drawn from a uniform random distribution over the range [0, 100]. All players know that the values are uniformly distributed.

ii. All the bots who wish to participate in the round need to submit a bid along with their confidence score. Confidence score signifies the bot's confidence of winning the auction round and lies between [0.5, 1].

iii. The bots who do not wish to participate in that round, can submit their bid = 0 and confidence score = 0.

iv. The payoff for the winner is given by **Payoff = c_i * (x_i – bid_i)**; where bid_i is the bid of the ith bot and c_i is the confidence score of ith bot.

v. All other bots, which participated in the round but did not win the round, receive a payoff defined by **Payoff = – (c_i * abs(x_i – bid_i))/10;** where abs(y) is the absolute value of y

vi. Bots which submitted bid = 0 and confidence = 0 (i.e. did not participate in that round), receive a **Payoff = 0.**

Variation 3 - Precision Matters

i. The value x_i for each player is drawn from a uniform random distribution over the range [0, 100]. All players know that the values are uniformly distributed.

ii. The payoff for the winner is given by
Payoff = (X - bid_i) - abs(min((bid_i - s_i), (X - bid_i)));
where X is the max value amongst all x_i for that round, bid_i is the winning bid, s_i is the second highest bid and min(a, b) is the minimum of a and b.

iii. In addition to this, the second-highest bidder will have to pay (50%) of what the winner earned (or lost) i.e. for second-highest bidders, payoff is given by
Payoff = - 0.5 * abs((X - bid_i) - (bid_i - s_i)); where abs(y) is the absolute value of y.

iv. All other players receive **Payoff = 0.**

Objective:

The objective of the game is to maximise the profit over t (~10^3) rounds. Participants must design a strategy for their bot to decide on the optimal bid based on the information available during each round.

Additional Details:

i. A starter code will be provided soon. Detailed walk through of the starter code will be done in the **orientation session on 31st May, 2025 at 9 pm**.
ii. Each player will only know their own x_i in each round, not the "value" of other players.
iii. Make sure that your bot does not take too long per round to run. If that happens, it may be removed from the auction. 1 second is the upper bound to have in mind.

iv. Additionally, if it is noticed that the bot is hogging up a lot of memory, it will be discarded from the auction. Try to limit memory usage to below 100 mB.
v. You may use any python library available. Explicitly mention the python libraries used in your report.
vi. Around 3 days before the deadline, we will hold a mock auction, where you can choose to submit your codes, and we will run a mini-auction on them. Relevant statistics such as net profit, variation of capital over time will be shared with you. The exact date of the mock auction will be announced on the Guild Whatsapp group.

Evaluation Criteria:

i. Total Profit will not be the only criteria for selections.
ii. Innovative ideas, logical basis for your strategy, as well as quality of the report will be judged for the selections.
iii. Robustness of your strategy to different starting capitals, number of positive payoff rounds and how your strategy adapts to different conditions will also be taken into account.

Resources:

Following is a list of non-exhaustive resources for the above Problem set. Introduction to Random Variables (Probability):
https://www.investopedia.com/terms/r/random-variable.asp

Introduction to Python Programming:
https://www.w3schools.com/python/python_intro.asp

Numpy, a useful python module:
https://www.w3schools.com/python/numpy/numpy_intro.asp

Deliverables:

Participants must submit their bot codes (.py files), which follows the provided template, along with a detailed report explaining their strategy.

Attach any relevant links, research papers and any other resources you went through for this application as well and a brief on how you applied the learnings from them.

A separate code is required for each variation. If your Rollno is AB24C123, then the submission for variation 1 should be named AB24C123_1.py. Similarly, the other submissions should be named AB24C123_2.py, AB24C123_3.py. Add all python files to a folder called “Submissions” in your google drive, and submit the link for this folder in a google form, which will be circulated 2 days before the **deadline (11:59 pm, 10th June 2025).**

May the force be with you!