1. **Voting in multi-attribute domains**

- If the five people vote individually, the outcome will be (P, S). If everyone is hopeful about their preferences, they will vote based on their first preference. For the swimming pool, three people vote for P and two people for $-P$. For the squash, three people vote for S and two people for $-S$. Based on the majority rule, they will build both.

  The outcome is bad because most people indicate their first preference of only having one between the swimming pool and the squash while the result is that both of them are being built. In fact, (P, S) is the least preferred choice for four out of the five voters.

- For each preference list, we assign 4 points to the first choice, 3 to the second, and so on. Then, the Borda count gives the outcome of (P, $-S$), the option that accumulates the most points from the five voters.

	(P, $-S$)	($-P$, S)	($-P$, $-S$)	(P, S)
1	4	3	2	1
2	4	3	2	1
3	3	4	2	1
4	3	4	2	1
5	3	2	1	4
Sum	17	16	9	8

  The outcome is overall satisfactory because two of the five voters list it as the first choice and one more lists as the second.

- One of the problems with the Borda count is its impracticality when dealing with multiple issues. If people have n issues to vote on, there are 2^n combinations of the preferences, which challenges agents' patience to finish the ballet.

  There are many other problems with the Borda count. It is susceptible to manipulations and not independent of irrelevant choices. (http://math.crg4.com/borda.html)

- If the outcome of each vote is revealed before going to the next vote, the outcomes will be different depending on the order of the items being voted.

  If building the swimming pool is voted first, then the outcome is (P, $-S$). At first, three people will vote for P and two for $-P$, the same as in part (a). Then, however, after they see the outcome to be P, four of them will vote for $-S$ because they prefer (P, $-S$) to (P, S), and only one vote for S.

  If building the squash is voted first, then the outcome is ($-P$, S). Three people will vote for S and two for $-S$, and then four of them vote for $-P$ and one vote for P.

2. **Voting rules**

There are 11 candidates and 1679 incomplete preference lists in total. All candidates that express some indifference between candidates have been removed. For plural vote, we assign one vote for the first candidate on each preference list. For Borda count, we assign 10 votes to the first candidate on a preference list, 9 to the second, and so on if they exist on the list. For approval vote, we assign one vote to all candidates that appear on a preference list.For Instant runoff, we adopt the same way as plural vote but iterate to eliminate the candidate with least votes every time until only one remains.

The chart below shows the results using different voting mechanisms.

	1	2	3	4	5	6	7	8	9	10	11	Win
Plural vote	246	453	412	384	352	27	94	463	37	0	0	8
Instant runoff	0	991	0	0	0	0	0	1226	0	0	0	8
Borda count	11012	10923	11717	13221	12525	5339	8130	13348	6276	160	53	8
Approval vote	1514	1532	1535	1731	1609	1012	1291	1679	1127	31	11	4

The approval vote gives a different outcome than the other three ways. This reflects the importance of designing an appropriate voting mechanism to represent people's preference. The goal, however, could be hard to achieve because each of the voting mechanisms makes some sense in a context while having some disadvantages. For example, as the plural vote is the most intuitive one among the four ways, it does not take account of a voter's preferences other than the top choice. Also, within a certain range, they all reflect that Candidate 4 and 8 are popular because they both receive significant amount of votes.

3. **Giving back to those who pay**

- The two mechanisms are both incentive compatible because the dominant strategy of an agent is to bid the actual value of a computer. Observe that the formula in either way contains a_i or b_i that do not depend on the bid of agent i. Thus, when agent i raises the bid, it only increases the amount of money to pay but not the amount of money to receive. Therefore, the agent does not have incentive to raise the bid. Also, getting a computer is preferred for an agent because the agent still receives the money besides the computer. Because a lower bid will decrease the competitiveness of the agent in getting a computer, the agent will not bid lower than its evaluation either.

  Both mechanisms make positive transfers because agents always gain no less than their payments. In cases when the agent does not get a computer, the agent does not need to pay according to the VCG rules and will receive refunds, which makes the earn positive. In cases when the agent gets a computer, the agent pay its evaluation and will receive a refund of $\frac{m}{n} * v_{m+2}$ or $\frac{m}{n-m-1} * v_{m+2} - \frac{m(m+1)}{(n-m-1)(n-m-2)} * v_{m+3}$ in the two mechanisms, respectively. The utility of the agent before receiving the refund is non-negative, and thus turns positive as the refund is of positive values.

  They are all individually rational because agents always make a non-negative profit. In cases when an agent gets a computer, the agent receives both the computer and the refund, which over-weights the payment (shown above). In cases when an agent does not get a computer, the agent receives a refund at no cost.

- The VCG mechanism charges agents the amount of negative impact they have on other agents. The total VCG payment in the second stage is the sum of those whose bids affect a_i and b_i and those whose bid do not. For a particular agent i, only the $m+1$ employers with the true highest evaluations influences a_i or b_i.

  In the first mechanism, they give a total payment of

$$(m+1) * \frac{m}{n} * v_{m+2}$$

  according to the formula. The rest of the employers pay a total of

$$(n-m-1) * \frac{m}{n} * v_{m+1}.$$

  Summing up the two parts gives the answer

$$(m+1) * \frac{m}{n} * v_{m+2} + (n-m-1) * \frac{m}{n} * v_{m+1}.$$

  Similarly, in the second mechanism, the first $m+1$ agents who affect both a_i and b_i would pay a total of

$$(m+1) * (\frac{m}{n-m-1} * v_{m+2} - \frac{m(m+1)}{(n-m-1)(n-m-2)} * v_{m+3})$$

  according to the formula. The $m+2$th agent only changes b_i, and the payment from this agent is

$$\frac{m}{n-m-1} * v_{m+1} - \frac{m(m+1)}{(n-m-1)(n-m-2)} * v_{m+3}.$$

Then, the rest of the agents pay

$$(n-m-2)*(\frac{m}{n-m-1}*v_{m+1}-\frac{m(m+1)}{(n-m-1)(n-m-2)}*v_{m+2}).$$

Simply summing up the three parts gives a final result.

- To compare the two ways, we calculate the difference between the two total payments and set it to be positive,

  $$\frac{m(m+1)}{n}*(v_{m+2}-v_{m+1})+\frac{m(m+1)(m+2)}{(n-m-1)(n-m-2)}*v_{m+3}>0.$$

  Then, we solve the inequality and obtain

  $$v_{m+2}-v_{m+1}>\frac{n(m+2)}{(n-m-1)(n-m-2)}*v_{m+3},$$

  which serves as the condition when the first mechanism gives back more than the second mechanism.

  Because the inequality does not always hold true, there does not exist a mechanism that always return more funds. Nevertheless, because the right hand side of the inequality is more likely a small number, for most of the occurrences the inequality should be true and thus the first mechanism usually generates better result for the agents.