Slide 1.

Hello, dear students. That's the third laboratory work for algorithms and data structures. Our topic is sorting algorithms. We will be mostly focused on usage of these algorithms, rather on implementing of them. So, sorting algorithms from standard libraries can be used.

Slide 2.

Today we will analyze possible solutions of problem "Line fighting" from your previous homework, and problems "Median on the Plane" and "Country of Fools"

Slide 3.

Problem 2025 - "Line Fighting". You can go through the link to check full description. I will remind you the most important part

- There can be from 2 to k teams taking part in a competition, and there are n fighters altogether in all the teams. Before the competition starts, the fighters are divided into teams: each fighter becomes a member of exactly one team.
- Two fighters fight each other if they are members of different teams.
- Help the organizers to distribute fighters between teams so as to maximize the number of fights and output this number.

Slide 4.

This problem isn't hard to be implemented. But first of all, we need to find an answers for a couple of questions.

1. What is optimal number of teams? We can split fighters in different number of teams from 2 to k.
2. What is optimal way to distribute fighters between teams? Are all teams should be equal, or as different as possible.

Let's start to answer on these questions with investigation of case with only two teams. It is clear, that if there are n fighters total and one team has m fighters in the team, that's mean that other team has (n – m) fighters. And because each fighter of first team should have fight with each fighter of the second team, we have m(n – m) fights in total.

Slide 5.

Now we can check more complex case. How to calculate all fights, if we have multiple teams?

In the similar way, if i-th team has mi fighters, that's mean that all other teams have (n-mi) fighters. By condition of problem each fighter of i-th team should fight with each fighters of each teams, so we have total mi(n – mi) fights for this team.

Let’s summarize them. Please, note, that calculating by this way, we count each fight two times. For example, for fights between i-th and j-th team, we calculate them 2 times, when calculate all fights for i-th team and when calculate all fights for j-th team. You can see final formula in the slide.

Let’s simplify this formula and open the brackets. After this sum can be separated on two parts, where sum of mi is sum of fighters from all teams, so it is equal to n.

Back to previous questions of optimal number of teams and distribution of fighters between each team. We can see in this formula, that it depends on both of k and mi. It is easy to understand, that total amount of fights will be maximum, if the sum in formula will be smallest possible. So, it is question of minimization of sum of squares.

Probably, the answer how to do this, what should be value of k and each mi is obvious. It is better to take maximum available k and split fighters on teams of the same size.

Slide 6.

Next part is optional and requires more deeper knowledge of mathematical analysis.

We will try to find, which distribution is optimal, purely mathematically to see interaction between algorithms and math.

Basically, we need to maximize number of fights, so we need minimize sum of squares of each team fighters. Let’s name this function f, and it is a function of multiple arguments.

Also, we know that sum of all fighters is n, which is constraint for functions f() arguments.

To find minimum of function with multiple arguments with constraints, we can use method of Lagrange multipliers. We will not deeply check this method, just use it to find minimum.

This method based on creating Lagrange function or Lagrangian. The format of this function you can see in the slide.

Let’s create constraint function g() from original constraint. For this purpose, we need to move n to the left part.

On the next step we can substitute functions f() and g() to Lagrangian, should be noted, that lambda is Lagrange multiplier and has behavior of constant.

Slide 7.

Next step is calculating partial derivatives for all arguments of Lagrangian. As was mentioned earlier, lambda is just constant for calculating derivatives.

To find extremums we should equalize all partial derivatives to zero and solve system of equations. It is very similar to usual method of single argument function analysis to find extremums. We will solve this system relative to lambda and express all mi via lambda.

The result you can see in the slide, each mi is equal to half of lambda.

Slide 8.

Now we can apply result of solution of system to constraint. We get sum of k terms, each equal to half of lambda.

We can express lambda via n and k. So, lambda is equal to 2n divided on k.

Final step to find stationary points is to apply this result for lambda to previous solution of system. Now we have a stationary point for function f() with constraint for its arguments. Each mi should be equal to n divided on k.

Slide 9.

It is easy to check, that in this point function f() has its minimum

Now we have an answer to one of questions. The optimal distribution of fighters between teams should be equal (or as close as possible to be equal, in case when n can't be divided on k evenly).

But we still have question of how many teams we should create for fighters.

Let's return to function f(), which should be minimized for maximum number of total fights. We can apply optimal values of mi, which were found earlier and check, what happened with the function.

Now function f() depends only on k and can be investigated from point of its extremums.

Slide 10.

Function f(k) is square of n divided on k, where square of n is constant. In this case, k is always greater than zero, because it is possible number of teams, and, of course, it can't be negative.

Plot of this function is known. It is decreasing hyperbola and function is monotonic. So, the value of function is less, when k is greater.

The final answer is following. It is better to take maximum available k and split fighters on teams of the same size.

With this knowledge it is easy to implement source code to solve this problem. But without any investigation you can't be sure that this approach to distribute fighters is optimal.

Should be noted, that other explanations are also possible. For example, this problem can be interpreted as finding which k-partite graph on n vertices has maximum number of edges and calculate it for that graph.

Slide 11.

Problem 1207 - "Median on the Plane". You can go through the link to check full description. I will remind you the most important part

- There are N points on the plane (N is even). No three points lie on the same straight line. Your task is to select two points in such a way, that straight line they belong to divides the set of points into two equal-sized parts.

Slide 12

Median on the Plane is geometric problem. We have coordinate plane and a bundle of points. Very important note, that no three points lie on the same straight line.

Nevertheless, how this problem can be solved?

We have small hint, that topic of this problem is sorting. How to use sorting in this problem? Can points be sorted?

Slide 13.

Very often one of the key idea related to sorting points is sorting them by angle relative to some axis or line.

In case of our problem, for example, if we have point D on the axis Ox, and 3 points A, B and C, we can sort them by angles between axis Ox and lines to these points from point D. On image on the slide, if points is sorted by angle, we can easily find, that point B lies “between” points A and C related to point D. And if draw line from D to B, it will split all other points on 2 parts, as it required in the problem.

Please, note, that we have no need to solve issue, when 2 points will have the same angle, by description of problem there are no 3 points lies on the straight line.

But the first question is – how to calculate the angles? The answer is simple – use trigonometric functions. But there are several function. Which function is better to use?

Slide 14.

Well, probably, each trigonometric function can be used. But we want to create simplest solution without handling a lot of special cases, related to this function, like, points with no value, and other.

So, we have continuous and monotonic function. And obvious candidate to this role is tangent in the right half of coordinate plane. As you can see on unit circle on the slide, it grows from minus infinity to infinity, and what is also important, can be easily calculated. The only issue is that tangent of 90-degree angles doesn’t exist. It will be solved later.

Slide 15.

Let’s check following example. For point B with coordinates (b1, b2) we can draw a line OB, and calculate tangent of angle BOx between axis Ox and line BO. To calculate tangent in right triangle by its description we should divide side opposed to angle BOx on side adjacent to it.

Should be noted, that for point B tangent will be negative, while for point A it will be positive.

Slide 16.

But we still can have several issues with using tangent function.

As you can see all previous calculations of tangent was related to center of coordinate plane or point on the axis Ox. But what if we don’t have such points? Or what if such point isn’t suitable for us, because all other points should have greater or equal abscissa, than chosen points, otherwise we will have issues with tangent.

To solve this issue, we just can use simple method of “moving” coordinate plane center to point with least abscissa (most “left” point on coordinate plane). For this method we will recalculate coordinates of all other points relative to new center of coordinate plane.

Slide 17.

A simpler approach for the same method, is just use relative coordinates to chosen point without any recalculations.

Let’s check the example, how it should work, and why we should choose the most left point.

Let point A to be the most left point with coordinates ax and ay. We will draw imaginary line a through point A and parallel to axis Ox. This line will be used to form angle for calculating tangent.

Choose any other point. For example, point B with coordinates bx and by. We need to find tangent of angle BAa. For this purpose, we will subtract ay from by to calculate opposite side to angle and subtract ax from bx to calculate adjacent side of angle. After dividing first on the second, we’ve got a tangent of this angle.

Please, note, that because we choose the most left point to be base in calculation of angles, all other points lie to the right from this point (or have the same abscissa). It means, that all values of tangent will be as expected lie from minus infinity to infinity, as was mentioned earlier in unit circle.

Slide 18.

The second possible problem is calculating of tangent for 90- and -90-degree angles. In these points value of tangent doesn’t exist. In practice, it means, that denominator of fraction, which is used to calculate tangent, will be equal to zero. And we can’t divide by zero.

This case should be handled manually. It is enough to use great enough value for such points. Please, note, that value should be positive if by greater than ay (as in case on image) and negative otherwise.

Slide 19.

Now we know how to calculate tangent for all angles related to point. Next step is sorting all points except base by value of tangent.

The middle point in sorted array will be an answer. In case of example in the image, it is point R.

Should be noted, that this problem also can be solved with other trigonometric and inverse functions like arctangent for top half of coordinate plane and others. But the main idea will be the same as in method explained earlier.

Also, sorting can be replaced with selection algorithm (like quickselect) to improve asymptotic complexity, because searching for median of array is a problem of finding k^{th} order statistic, where k is size of array divided by two.

Slide 20.

Problem 1604 - “Country of Fools”. You can go through the link to check full description. I will remind you the most important part

- The chief traffic policeman ordered n speed limit signs. When the order arrived, it turned out that the signs had different numbers on them, which showed limits in kilometers per hour. There were k different limits: n_1 signs with the first limit, n_2 signs with the second limit, etc.; the total number of signs is n.
- The chief policeman decided to place the signs on the motorway in such a way that a driver would have to change speed as many times as possible. A speed limitation is valid until the following speed limit sign. For example, if there is the number 60 on the sign, then a car must go until the following sign with the speed of exactly 60 kilometers per hour.

Slide 21.

The description of problem requires, that signs in a row should have different types as much as possible. In other words, we should minimize number of times, when 2 consequent signs have the same type.

Let’s try the first idea – simply just use all types of signs in a loop. It perfectly works, when we have equal number of each types of signs. For example, 3 signs of 3 types, each type is mentioned with number. Output is sign of type one, the type 2, 3,1,2,3,1,2,3.

But when we have number of third type greater, than others, we will have a problem. For example, we have 2 signs of the first two types and 4 signs of the third type. Output is 1,2,3,1,2,3,3,3. We have 3 signs of third type in a row, but it is obvious, that we could have none, for example, 3,1,3,2,3,1,3,2.

Slide 22.

Let’s try another approach. On the slide you can see different number of signs of 5 types. Let’s sort them.

Slide 23.

What to do next? First of all, we should find a real problem of sign distribution. And this problem is type with greatest number of signs. If we don’t spend signs of this type – we will have problems in the end, because it will be impossible to find another type of sign to rotate with it. So, the most obvious choice – on each iteration use in rotation signs of 2 types with greater amount than other types.

Slide 24.

But wait. Or… maybe we can rotate type with greatest number of signs and… any other type?

To check it we should create a metric of quality of current state. What is bad situation for us? As was mentioned earlier, when we should output signs of the same type in a row.

So, let the number of signs of one type in a row be a metric of current state. Let’s name it “row length”. It is obvious, that after initial sorting we can say exactly, what is possible minimum of signs of one type in row can be.

As was explained earlier, the main problem is type with greatest number of signs. It can be noticed, that we can't avoid using the same type of signs in a row, if we have to many signs of one type. To be exactly, this amount is equal to sum of all other types plus 1. In this case we should print at least 2 signs of the same type in a row. The greater the difference between maximum type and sum of all other types, than more signs of the same type in a row we will have.

The formula to calculate "row length" can be seen on slide.

Slide 25.

Now we have a metric, so we should answer on several questions related to behavior of this metric during choosing signs.

First question is: can "row length" become better? And the answer is no, it can't by its description. It is already minimum possible number of signs of the same type in a row.

Second question. Can "row length" become worse? Of course, it can. If we make a bad choice. Let's remember first example of bad strategy, when we just use all types of signs in a loop. Initially "row length" should be equal to zero, but after a couple of choices of signs, it becomes equal to 2. That's a confirmation, that this strategy is bad.

Next question. What if "row length" doesn't change? It means that our strategy works very well! As we know, we can't make "row length" better. But we decrease total amount of signs, and don't make situation worse.

Based on this conclusion, it can be said, that "row length" is invariant, which must be satisfied by optimal strategy.

And the last question, what is behavior of "row length", if we choose on each iteration type with maximum number of signs and any other type? This question can be reformulated as, does this strategy satisfy invariant?

Let's check. On each iteration we output 2 signs – from type with maximum number of signs and random one. It means, that on each iteration we decrease type with maximum number of signs by one. But also, on each iteration we decrease sum of all other types by 1. Invariant is satisfied. This strategy is optimal.

Should be noted, that there is no need to implement this random strategy. For example, on each iteration you can output 2 types with the greater number of signs than other, or rotate types with maximum and minimum number of signs.

Slide 26.

Let's check another possible solution. After initial sorting we can split all signs of type with maximum amount into buckets. So, we have number of buckets equal to number of type with maximum number of signs.

In the image we add type 2 to the buckets.

Slide 27.

Start from first bucket let split all other types of signs in order of decreasing of their amount.

Firstly, we will add type 3 as next type with maximum amount

Slide 28.

Secondly, we will add type 5.

Note, that for next type of sign you shouldn't start again from first bucket, but continue from bucket with less signs

Slide 29.

And finally, we can add signs of type 1 and 4.

Why is this method working?

Because each bucket always contains only unique types of sign. After this we can output buckets one by one.

Should be noted that there is only one case, when we can't avoid using 2 signs of the same type in a row – when number of this type is greater than sum of all other types. That's mean, that we have the same conclusion as for previous approach to solve this problem.

Slide 30.

Mandatory task. You should implement source code for problem 1207 "Median on the Plane", upload it to Timus system and pass all tests. After this you should prepare a report and send it via email. Please, use template document for your report and carefully set correct subject for the mail.

Slide 31.

You will have 2 problems for homework. First problem is 1604 "Country of Fools", which was explained earlier. Second problem is 1444 "Elephpotamus". You should solve it by yourself. Please, note, that report for this problem should contain explanation of your solution and answer to question: what is the difference with problem #1207 "Median on the Plane"?

Homework is optional, but successful completion of this problems can give you extra points for better grade.

Slide 32.

Thank you for attention