Programming Arena ver1.0

Tutorial

UNIVERSITY OF AIZU 1

1. What is Programming Arena?

AOJ Arena	× +			- 🗆 ×
\leftrightarrow \rightarrow C $$ or	linejudge.u-aizu.ac.jp/beta/arena.html			ञ २ 🛧 🔞 🖸
	Arena Arenas - Coordinate		aizu_z Logout Sign Up	
	Programming Arena			
	This is the β version of Arena for Aizu Online Judge 2.0.			
	CONTEST 🗹	VIRTUAL 🖻	EXERCISE 🕑	
	Updated: 2019/09/10. Please clear the cash of your browser. If you find any problems, please let us know (uosojs@gmail.com). Thank you	ivery much for your cooperations		
	п учи них ану ичинена, реаве не из кном (човородулал. сонт). Паше учи	тету плася по учил соврегацията.		
	Aizu Online Judge 2.0 API		Page Top	
	Azu onime Sudge 2.0 API		Page top	
				*

https://onlinejudge.u-aizu.ac.jp/beta/arena.html

Programming Arena is a tool to organize arenas each of which consists of a set of problem from Aizu Online Judge (AOJ). There are three types of arenas as follows:

- **Exercise** : Oriented towards introduction to programming and algorithms which use elementary or typical problems from Courses
- Virtual : Oriented towards virtual programming contests witch use past problems from Challenge
- Contest : Oriented towards real programming contests which use original problem sets

There are three roles in Arena as follows:

- Participants : Users who solve problems in Arenas. Students and competitors.
- **Coordinators** : Users who create and manage Exercises and Virtuals. Teachers and coaches.
- Manager : The admin who can create and manage Contests. (* now AOJ admin)

2. Entrance of Arena

You can browse a list of Arenas from the top page of Programming Arena. Generally, participants enter an Arena via URL given from its Coordinator.

AOJ Arena	× +								-		>
→ C 🔒 o	onlinejudge.u-aizu.ac.jp/beta/arena.htm	l#exercise						07	Q	☆ (1	
	Arena Arenas - Coordinate					ł	aizu_z Logout Sign Up				
	Running										
	#	Title	Start	End	Coordinator	Info	Policy State				
	Scheduled										
	#	Title	Start	End	Coordinator	Info	Policy State				
	Ended										
	#	Title	Start	End	Coordinator	Info	Policy State				
	UoA_ALDS1_2019_C34_13	University of Aizu Algorithms and Data Structures 2019 Topic 13	2020/1/31 13:20:00	2020/2/7 13:20:00	F0118T		public Finished				
	UoA_ALDS1_2019_C12_13	University of Aizu Algorithms and Data Structures 2019 Topic 13	2020/1/30 15:10:00	2020/2/6 15:10:00	F0118T		public Finished				
	UoA_ALDS1_2019_C56_13	University of Aizu Algorithms and Data Structures 2019 Topic 13	2020/1/30 15:10:00	2020/2/6 15:10:00	F0118T		public Finished				
	UoA_ALDS1_2019_C34_12	University of Aizu Algorithms and Data Structures 2019 Topic 12	2020/1/29 13:20:00	2020/2/5 13:20:00	F0118T		public Finished				
	UoA_ALDS1_2019_C34_11	University of Aizu Algorithms and Data Structures 2019 Topic 11	2020/1/28 13:20:00	2020/2/4 13:20:00	F0118T		public Finished				

3. Activities in Arena: Participants

Login and Entry

Home provides a basic information of the arena. First of all, users need to login to Programming Arena with AOJ ID. Then a participant has to enter the Arena through ENTRY to solve problems. If the Arena is private, he/she must input its password given by the Coordinator.

Aizu Online Judge Arena 🗙	+ -		×
\leftrightarrow \rightarrow \mathcal{C} $($ a onlinejudge.u-aizu	acjp/beta/room.html#UoA_ALDS1_2019_C12_02 💁 🤤 🙀	1112	0
University of Aizu	Algorithms and Data Structures 2019 Topic 2 Back to Arena Lang * Login		
2019/12/9 15:10:00	2019/12/16 15:10:00		
A Home Problems I Works	pace \equiv Submissions $i^{ m F}$ Ranking i Board ${ m tel}$ Analysis A Rejudge	↓F	
ID:	UoA_ALDS1_2019_C12_02		
Title:	University of Aizu Algorithms and Data Structures 2019 Topic 2		
Start Time:	2019/12/9 15:10:00		
End Time:	2019/12/16 15:10:00		
Current Time:	2020/4/8 19:42:44		
Coordinator:	F0118T		
Information:			
Comments:			
State:	Finished		
	Lentry		

Problem

You can see a list of problems given by the arena.

→ C	ac.jp/beta/room.html#UoA_ALDS1_2019	9_C12_01/problems		or Q 🕁
University of	Aizu Algorithms and Data Struct	ures 2019 Topic 1 Back to Arena	🧶 alzu_z 💡	Japanese - Logout
19/12/5 15:10:00		2128:06:07		2020/7/2 15:10:00
🕈 Home 🌓 Problems 🝼 Works;	ace ≡ Submissions ↓ ^F Ranking	ί Board 🛄 Analysis 🔺 Rejudge		
A L	в	c 🗵	D	
Insertion Sort	Greatest Common Divisor	Prime Numbers	Maximum Profit	
	▲× 63 🖌	≜ × 44 √	≗ × 18 ✓	
≗×65 ◀				

You can click a panel (or link) to browse the problem description.

t Home 🏠 Problems 🖌 Workspace ≡ Submissions JF Ranking i Board 🔛 Analysis 🛦 Rejudge	izu Online Judge Arena	× +																																															-				
M25 15:100 2020 2020 2020 2020 2020 2020 2020	→ C	judge.u-aizu.ac.jp/b	eta/room.html#U	oA_ALDS1_201	19_C12_01/p	oroblems/A																																								0	-	e	ξ.	☆	(有陸)
Home Problems Workspace Esubmission Fanking Long A Rejudge Problem A Tracturet ise Memory Lett 1907:00 Image: Source and Source	U	niversity of Aizu	Algorithms and	d Data Struc	tures 201	9 Topic 1	Back to Arer	na																											4	<u>e</u> a	izu	Z	•	Jap	ines	e▼	Lo	igol	ıt								
Home Problems Y Workspace E Submission I F Ranking E Manakyis A Rejudge Problem A TimeLine: 1	/12/5 15:10:00					2128:0	6:07																																						2	020/7	7/2	15	:10:0	10			
Product A may Link 1987 00 may Link	Home Problems	✓ Workspace	≡ Submissions	JF Ranking	i Board	🛃 Analysis	A Rejudg	je																																												ţ	
whule i financial field of the set of the s		nit: 131072 KB																																												2						1	
whule i financial field of the set of the s																																																					
5場合、1枚ずつカードを取り出して、それをその時点ですでにソートされている並びの適切な位置に挿入していくことによって、カードを並べ替えることができます。 #入ソートは次のようなアルゴリズムになります。 1 insertionSort(A, N) // N個の要素を含む0-オリジンの配列A 2 for i が 1 から N-1 まで 3 v = A[1] 4 j = i - 1 5 while j >= 0 かつ A[j] > v 6 A[j+1] = A[1] 7 j 8 A[j+1] = v V 個の要素を含む数列 A を昇順に並び替える挿入ソートのプログラムを作成してください。上の疑似コードに従いアルゴリズムを実装してください。アルゴリズムの動作を確認す 5 ため、各計算ステップでの配列(入力直後の並びと、各 i の処理が終了した直後の並び)を出力してください。 入力 入りの最初の行に、数列の長さを表す整数 N が与えられます。2 行目に、N 個の整数が空白区切りで与えられます。	挿入ソート																																																				
5명合、1枚ずつカードを取り出して、それをその時点ですでにソートされている並びの適切な位置に挿入していくことによって、カードを並べ替えることができます。 年入ソートは次のようなアルゴリズムになります。 1 insertionSort(A, N) // N個の要素を含む0-オリジンの配列A 2 for i が 1 から N-1 まで 3 v = A[i] 4 j = i - 1 5 while j >= 0 かつ A[j] > v 6 $A[j+1] = k[j]$ 7 j 8 $A[j+1] = v$ 4 Gの要素を含む数列 A を昇順に並び替える挿入ソートのプログラムを作成してください。上の疑似コードに従いアルゴリズムを実装してください。アルゴリズムの動作を確認す 5 ため、各計算ステップでの配列(入力直後の並びと、各 i の処理が終了した直後の並び)を出力してください。 大力																																																					
 ・(A, N) // N個の要素を含むe・オリジンの配列A for i が 1 から N-1 まで v = A[i] j = i - 1 second (i = 1) v = A[i] j = i - 1 second (i = 1) v = A[i] d [j = i - 1 second (i = 1) v = A[i] v v = A[i] v = A[i] v v @ o 要素を含む数刻 A を用順に並び替える(i = 0) v = Dia v = Dia v = Dia v v @ o 要素を含む v = Dia v = Dia v = Dia v v Moo dia v = Dia v = Dia v = Dia v v Moo mon v = A[i] v v Moo m v = Dia v = Dia v v Moo mult v = Dia v v Moo v v Moo	F3.)/ b (Turnet	(++	₩7(ŦŦ ⊃ _ L	+ (- /# 10 10	マーウキーの		h			71	71			 									_	_		-					-		-	-+	+				-1-	.+-	44.0		+		×1	**	~					
1 insertionSort(A, N) // N個の要素を含む8-オリジンの配列A 2 for i が 1 から N-1 まで 3 v = A(i) 4 j = i - 1 5 while j >= 0 かつ A(j) > v 6 A(j+1) = A(j) 7 j 8 A(j+1) = v V 個の要素を含む数列 <i>A</i> を昇順に並び替える挿入ソートのプログラムを作成してください、上の疑似コードに従いアルゴリズムを実装してください。アルゴリズムの動作を確認す るため、各計算ステップでの配列(入力直後の並びと、各 <i>i</i> の処理が終了した直後の並び)を出力してください。 入力 入りの最初の行に、数列の長さを表す整数 <i>N</i> が与えられます。2 行目に、 <i>N</i> 個の整数が空白区切りで与えられます。																																													L VII	頁(こ)	並	ベ					
<pre>2 for i が 1 から N-1 まで 3 v = A[1] 4 j = i - 1 5 while j >= 6 かつ A[j] > v 6 A[j+1] = A[j] 7 j 8 A[j+1] = v N 個の要素を含む数列 A を昇順に並び替える挿入ソートのプログラムを作成してください。上の疑似コードに従いアルゴリズムを実装してください。アルゴリズムの動作を確認す るため、各計算ステップでの配列(入力直後の並びと、各 i の処理が終了した直後の並び)を出力してください。 入力 入力の最初の行に、数列の長さを表す整数 N が与えられます。2 行目に、N 個の整数が空白区切りで与えられます。</pre>																																														頁(こ)	並	~					
<pre>2 for i が 1 から N-1 まで 3 v = A[1] 4 j = i - 1 5 while j >= 6 かつ A[j] > v 6 A[j+1] = A[j] 7 j 8 A[j+1] = v V 級の要素を含む数列 A を昇順に並び替える挿入ソートのプログラムを作成してください。上の疑似コードに従いアルゴリズムを実装してください。アルゴリズムの動作を確認す るため、各計算ステップでの配列(入力直後の並びと、各iの処理が終了した直後の並び)を出力してください。 入力 入力の最初の行に、数列の長さを表す整数 N が与えられます。2 行目に、N 鶴の整数が空白区切りで与えられます。</pre>	る場合、1枚ずつ力	ードを取り出して	それをその時点																																											頁(こ)	並	~					
 y = A[i] j = i - 1 while j > e かつ A[j] > v A[j+1] = A[j] J A[j+1] = v N @の要素を含む数列 A を昇順に並び積える挿入ソートのプログラムを作成してください、上の疑似コードに従いアルゴリズムを実装してください。アルゴリズムの動作を確認するため、名計算ステップでの配列 (入力直後の並びと、各 i の処理が終了した直後の並び)を出力してください。 入力の最初の行に、数列の長さを表す整数 N が与えられます。2 行目に、N 個の整数が空白区切りで与えられます。 	る場合、1枚ずつ力	ードを取り出して	それをその時点																																											頁(こ)	並	~					
4 j = 1 · 1 5 while j = 0 かつ A[j] > v 6 A[j+1] = A[j] 7 j 8 A[j+1] = v N 個の要素を含む数列 A を昇順に並び増える挿入ソートのプログラムを作成してください。上の疑似コードに従いアルゴリズムを実装してください。アルゴリズムの動作を確認す るため、各計算ステップでの配列(入力直後の並びと、各 i の処理が終了した直後の並び)を出力してください。 入力 入力の最初の行に、数列の長さを表す整数 N が与えられます。2 行目に、N 個の整数が空白区切りで与えられます。	る場合、1枚ずつ力 挿入ソートは次のよ	ードを取り出して うなアルゴリズム	、それをその時点 こなります。	ですでにソー																																										頁(こ)	並	~					
5 ihile j >= 0 がつ A[j] > v 6 A[j+1] = A[j] 7 j 8 A[j+1] = v N Mor要素を含む数列 A を昇順に並び替える挿入ソートのプログラムを作成してください。上の疑似コードに従いアルゴリズムを実装してください。アルゴリズムの動作を確認するため、名計算ステップでの配列 (入力直後の並びと、名 i の処理が終了した直後の並び)を出力してください。 入力の最初の行に、数列の長さを表す整数 N が与えられます。2 行目に、N 個の整数が空白区切りで与えられます。	る場合、1枚ずつ力 挿入ソートは次のよ 1 insertionSort(/	ードを取り出して うなアルゴリズム A,N) // N個の要素	、それをその時点 こなります。	ですでにソー																																										頁(こ)	並	~					
6 A[j+j] = A[j] 7 j 8 A[j+j] = v N 個の要素を含む数列 A を昇順に並び替える挿入ソートのプログラムを作成してください。上の疑似コードに従いアルゴリズムを実装してください。アルゴリズムの動作を確認す るため、名計買ステップでの配列(入力直後の並びと、名 <i>i</i> の処理が終了した直後の並び)を出力してください。 入力 入力の最初の行に、数列の長さを表す整数 N が与えられます。2 行目に、N 個の整数が空白区切りで与えられます。	る場合、1枚ずつ力 挿入ソートは次のよ 1 insertionSort(/ 2 for i が 1 か 3 v = A[i]	ードを取り出して うなアルゴリズム A,N) // N個の要素	、それをその時点 こなります。	ですでにソー																																										頁(こ)	並	~					
7 j・ 3 8 A[j+1] = v N 個の要素を含む数列 A を昇順に並び替える挿入ソートのプログラムを作成してください。上の疑似コードに従いアルゴリズムを実装してください。アルゴリズムの動作を確認す るため、各計算ステップでの配列(入力直後の並びと、各 i の処理が終了した直後の並び)を出力してください。 入力 入力の最初の行に、数列の長さを表す整数 N が与えられます。2 行目に、N 個の整数が空白区切りで与えられます。	る場合、1枚ずつ力 挿入ソートは次のよ 1 insertionSort(A 2 for i が 1 か 3 v = A[i] 4 j = i - 1	ードを取り出して うなアルゴリズム A, N) // N個の要素 ら N-1 まで	、それをその時点 こなります。	ですでにソー																																										頁(こ)	並	~					
8 A[j+1] = v N 個の要素を含む数列 A を昇順に並び構える挿入ソートのプログラムを作成してください。上の疑似コードに従いアルゴリズムを実装してください。アルゴリズムの動作を確認するため、含計算ステップでの配列(入力直後の並びと、各 i の処理が除了した直後の並び)を出力してください。 入力 入力の最初の行に、数列の長さを表す整数 N が与えられます。2 行目に、N 個の整数が空白区切りで与えられます。	る場合、1枚ずつ力 挿入ソートは次のよ 1 insertionSort(A 2 for i が 1 か 3 v = A[i] 4 j = i - 1 5 while j >=	ードを取り出して うなアルゴリズム A, N) // N個の要素 ら N-1 まで 8 かつ A[j] > v	、それをその時点 こなります。	ですでにソー																																										頁(こ)	並	~					
N 個の要素を含む数列 A を昇順に並び替える挿入ソートのプログラムを作成してください。上の疑似コードに従いアルゴリズムを実装してください。アルゴリズムの動作を確認す るため、各計算ステップでの配列(入力直後の並びと、各 i の処理が終了した直後の並び)を出力してください。 入力 入力の最初の行に、数列の長さを表す整数 N が与えられます。2 行目に、N 個の整数が空白区切りで与えられます。	る場合、1枚ずつ力 挿入ソートは次のよ 1 insertionSort(/ 2 for i が 1 か 3 v = A[i] 4 j = i - 1 5 while j >= 6 A[j+1] =	ードを取り出して うなアルゴリズム A, N) // N個の要素 ら N-1 まで 8 かつ A[j] > v	、それをその時点 こなります。	ですでにソー																																										頁(こ)	並	~					
るため、各計算ステップでの配列(入力直後の並びと、各 i の処理が終了した直後の並び)を出力してください。 入力 入力の最初の行に、数列の長さを表す整数 N が与えられます。2 行目に、N 個の整数が空白区切りで与えられます。	る場合、1枚ずつ力 挿入ソートは次のよ 1 insertionSort(/ 2 for i が 1 か 3 v = A[i] 4 j = i - 1 5 while j >= 6 A[j+1] = 7 j	ードを取り出して うなアルゴリズム A, N) // N個の要素 ら N-1 まで 8 かつ A[j] > v	、それをその時点 こなります。	ですでにソー																																										頁(こ)	並	~					
入力 入力の最初の行に、数列の長さを表す整数 N が与えられます。2 行目に、N 働の整数が空白区切りで与えられます。	る場合、1枚ずつ力 挿入ソートは次のよ 1 insertionSort(/ 2 for i が 1 か 3 v = A[i] 4 j = i - 1 5 while j >= 6 A[j+1] = 7 7 j 8 A[j+1] = v	ードを取り出して うなアルゴリズム A, N) // N個の要素 ら N-1 まで e かつ A[j] > v A[j]	それをその時点 こなります。 を含む0-オリジン	iですでにソー の配列A	-トされてい	る並びの適	切な位置に持	挿入し				,			 							C				,1<	<2	221	CJ	5	τ、	力)—	4	E i	並	「「「「」」	え	20	:2:	55.00	:53	ŧ₫	•									
入力の最初の行に、数列の長さを表す整数 N が与えられます。2 行目に、N 個の整数が空白区切りで与えられます。	る場合、1枚ずつ力 挿入ソートは次のよ 1 insertionSort(A 2 for i が 1 か 3 v = A[i] 4 j = i - 1 5 while j >= 6 A[j+1] = v N 個の要素を含む数	 ドを取り出して うなアルゴリズム A, N) // N個の要素 ら N-1 まで e かつ A[j] > v A[j] 例 A を昇順に並び 	 それをその時点 こなります。 を含む0・オリジン <	ですでにソー の配列A トのプログラ:	- トされてい ムを作成し	いる並びの適! てください。	切な位置に打	■入し -ド	10	5 5 6			(2	-			(2						1	(1)	従	,ヽ< Éしヽ	<2	221	CJ	5	τ、	力)—	4	E i	並	「「「「」」	え	20	:2:	55.00	:53	ŧ₫	•									
入力の最初の行に、数列の長さを表す整数 N が与えられます。2 行目に、N 個の整数が空白区切りで与えられます。	る場合、1枚ずつ力 挿入ソートは次のよ 1 insertionSort(A 2 for i が 1 か 3 v = A[i] 4 j = i - 1 5 while j >= 6 A[j+1] = v N 個の要素を含む数	 ドを取り出して うなアルゴリズム A, N) // N個の要素 ら N-1 まで e かつ A[j] > v A[j] 例 A を昇順に並び 	 それをその時点 こなります。 を含む0・オリジン <	ですでにソー の配列A トのプログラ:	- トされてい ムを作成し	いる並びの適! てください。	切な位置に打	■入し -ド	10	5 5 6			(2	-			(2						1	(1)	従	,ヽ< Éしヽ	<2	221	CJ	5	τ、	力)—	4	E i	並	「「「「」」	え	20	:2:	55.00	:53	ŧ₫	•									
	る場合、1枚ずつ力 挿入ソートは次のよ 1 insertionSort(/ 2 for i が 1 か 3 v = A[i] 4 j = i - 1 5 while j >= 6 A[j+1] = 7 7 j 8 A[j+1] = v N 個の要素を含む器 るため、名計算ステ	 ドを取り出して うなアルゴリズム A, N) // N個の要素 ら N-1 まで e かつ A[j] > v A[j] 例 A を昇順に並び 	 それをその時点 こなります。 を含む0・オリジン <	ですでにソー の配列A トのプログラ:	- トされてい ムを作成し	いる並びの適! てください。	切な位置に打	■入し -ド	10	5 5 6			(2	-			(2						1	(1)	従	,ヽ< Éしヽ	<2	221	CJ	5	τ、	力)—	4	E i	並	「「「「」」	え	20	:2:	55.00	:53	ŧ₫	•									
	る場合、1枚ずつ力 挿入ソートは次のよ 1 insertionSort(/ 2 for i が 1 か 3 v = A[i] 4 j = i - 1 5 while j >= 6 A[j+1] = 7 7 j 8 A[j+1] = v N 個の要素を含む器 るため、名計算ステ	 ドを取り出して うなアルゴリズム A, N) // N個の要素 ら N-1 まで e かつ A[j] > v A[j] 例 A を昇順に並び 	 それをその時点 こなります。 を含む0・オリジン <	ですでにソー の配列A トのプログラ:	- トされてい ムを作成し	いる並びの適! てください。	切な位置に打	■入し -ド	10	5 5 6			(2	-			(2						1	(1)	従	,ヽ< Éしヽ	<2	221	CJ	5	τ、	力)—	4	E i	並	「「「「」」	え	20	:2:	55.00	:53	ŧ₫	•									
出力	 る場合、1枚ずつ力 挿入ソートは次のよ 1 insertionsort(/ 2 for iが 1か 3 v = A[t] 4 j = i - 1 5 while j >= 6 A[j+1] = r 8 A[j+1] = v N 個の要素を含む認 るため、各計算ステ 入力 	 ドを取り出して うなアルゴリズム み、N) // N個の要素 ら N-1 まで e かつ A[j] > v A[j] 例 A を昇順に並び ップでの配列(入 	 それをその時点 こなります。 を含む0-オリジン /皆える挿入ソー /街直後の並びと、 	iですでにソー の配列A トのプログラ 各 <i>i の</i> 処理か	ムを作成し 総容した値	へる並びの適 てください。 「後の並び):	」 上の疑似⊐ を出力してぐ	Ⅰ −ド くださ	ドルごさ	じょさ	 こ こ	(c ±		2		12		12	12	2					従い。	, ヽ < Éしヽ	< <i>2</i>	221	CJ	5	τ、	力)—	4	E i	並	「「「「」」	え	20	:2:	55.00	:53	ŧ₫	•									
	 る場合、1枚ずつ力 挿入ソートは次のよ 1 insertionsort(/ 2 for iが 1か 3 v = A[t] 4 j = i - 1 5 while j >= 6 A[j+1] = r 8 A[j+1] = v N 個の要素を含む認 るため、各計算ステ 入力 	 ドを取り出して うなアルゴリズム み、N) // N個の要素 ら N-1 まで e かつ A[j] > v A[j] 例 A を昇順に並び ップでの配列(入 	 それをその時点 こなります。 を含む0-オリジン /皆える挿入ソー /街直後の並びと、 	iですでにソー の配列A トのプログラ 各 <i>i の</i> 処理か	ムを作成し 総容した値	へる並びの適 てください。 「後の並び):	」 上の疑似⊐ を出力してぐ	Ⅰ −ド くださ	ドルごさ	じょさ	 こ こ	(c ±		2		12		12	12	2					従い。	, ヽ < Éしヽ	< <i>2</i>	221	CJ	5	τ、	力)—	4	E i	並	「「「「」」	え	20	:2:	55.00	:53	ŧ₫	•									

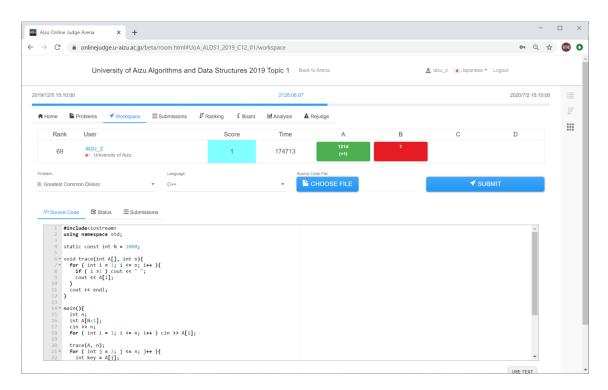
AOJ Arena Tutorial

2 3		
力例 2 に対する出力		
2 3 2 3		
2 3		
Note		
Algorithm		
	SUBMIT	

If the Arena is EXERCISE, links to commentary are available (if any) within the problem description.

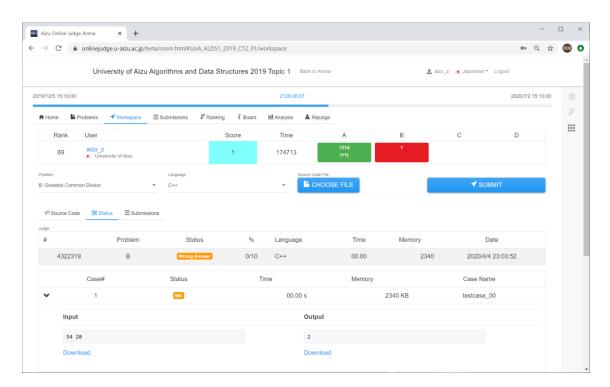
You can move to Workspace with the corresponding problem ID through SUBMIT button.

Workspace



The Workspace is to submit source codes and check verdict for the selected problem. You can see your progress state at the top of the Workspace.

When you submit your source code, please confirm the problem ID and programming language you use. Select the source code from CHOOSE FILE then the source code is shown in the Source Code area. Then, you can submit the code by SUBMIT button.



When the verdict becomes available, the judge result is shown in the Status.

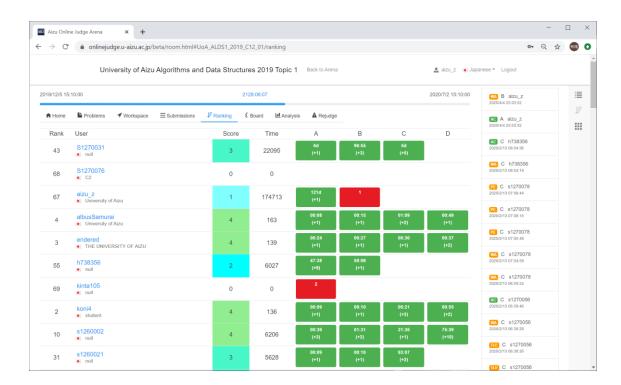
You can browse test cases (judge data) from SHOW DETAILS.

Here are the details of judge status.

Result	Details of the status
CE COMPILE ERROR	Compiler the judge uses has failed to compile your program. Please see error messages from the judge. If you need, please see the version of compilers at System Information.
RUNTIME ERROR	Your program has failed during the execution. Possible causes include: out of range in pointer reference (Segmentation Fault), stack overflow (infinite recursive functions), division by zero and many others. Note that, the main function in C language should return 0.
TLE TIME LIMIT EXCEEDED	The CPU time your program has run has exceeded the time limit specified by the corresponding problem. Note that, the judge runs your program up to around the Time Limit +1 seconds, and the judge may be forcibly terminated around 40 seconds with Runtime Error.
MLE MEMORY LIMIT EXCEEDED	The maximum amount of memory that your program has used has exceeded the limit specified for the corresponding problem.
OLE OUTPUT LIMIT EXCEEDED	Your program has produced too much output.
WA WRONG ANSWER	Your program has produced the output which is different from the judge data. In the case of special judges, the judge may return Wrong Answer depending on results of validators prepared for the corresponding problem.
PE PRESENTATION ERROR	Your program is almost accepted but not perfect. Your program outputs extra spaces or blank lines, and/or does not output required spaces or blank lines.
AC ACCEPTED	Your program has been "accepted" since it has not failed into all the above mentioned conditions.

Ranking

You can browse status of all participants through the leaderboard. The participants can be sorted by their scores or IDs.



Board

Board is for Q&A and clarification between participants and coordinators.

	Kyoto L	University Programming Contest	2020 Spring Back to Arena		•	F0118T	Japanese 👻 Logout	
A Home	Problems	s ✔Workspace ≡Submissions	J [∓] Ranking i Board ⊡ Analysis	A Rejudge				
Thread Li	st							
#	Problem	Title		User	Туре	Policy	Date	
688				nvip62	question	public	2020/3/20 12:52:27	
689	Ν			kupc_official	announce	public	2020/3/20 13:12:14	
690	D	D 問題のテストケース不備に伴うリ	ジャッジについて	kupc_official	announce	public	2020/3/20 14:38:41	
693	Ν	与えられる入力はランダムではなく	、恣意的である場合がありますか?	team_tsukukoma	question	public	2020/3/20 16:09:16	
			CREATE THREAD					