Компьютерные инструменты в образовании, 2018
№ 6: 47–55
УДК: 004.942:531.01
http://ipo.spb.ru/journal
doi:10.32603/2071-2340-2018-6-47-55

ПРИНЦИПЫ ФОРМИРОВАНИЯ ОЛИМПИАДНОЙ КОМАНДЫ ПО ПРОГРАММИРОВАНИЮ

Корабельщикова С. Ю.[1], Толкачева Е. А.[2], Бутин К. П.[1]

[1]Северноый (Арктичесий) федеральный университет им. М. В. Ломоносова, Архангельск, Россия
[2]Санкт-Петербурский Государственный Электротехнический Университет «ЛЭТИ» им. В. И. Ульянова (Ленина), Санкт-Петербрг, Россия

Аннотация

Рассмотрен вопрос организации команды и тренировочного процесса для участия в олимпиадах по командному программированию. Разработаны принципы формирования состава сборной, роли каждого из участников команды, исходя из психологических особенностей мышления и уровня сформированности компетенций. Выбрана классификация видов мышления на основе стандартности-нестандартности решаемых задач и операциональных процедур мышления. Нередко в олимпиадах по программированию встречаются задачи, которые можно свести к задаче о рюкзаке. Приведена система тренировочных заданий, сводящихся к задаче о рюкзаке. Задания расположены по нарастанию сложности. Сначала рассматривается задача о неограниченном рюкзаке и различные алгоритмы её решения. Для частного случая данной задачи, когда стоимость предмета равна весу предмета, представлены алгоритм и различные его реализации. Разработанная система тренировок и принципы формирования команды позволили достигнуть высоких результатов, в частности пройти внутренние отборочные этапы и достойно выступить в чемпионате мира по командному программированию, что подтверждает эффективность предложенных методов.

Ключевые слова: *олимпиада по программированию, командная олимпиада, виды мышления, система тренировок, компетенции высшего профессионального образования, задача о рюкзаке.*

Цитирование: Корабельщикова С. Ю., Толкачева Е. А., Бутин К. П. Принципы формирования олимпиадной команды по программированию // Компьютерные инструменты в образовании, 2018. № 6. С. 47–55. doi:10.32603/2071-2340-2018-6-47-55

1. ВВЕДЕНИЕ

Для когнитивного процесса личности необходимым толчком является конфликтная ситуация между данностью и необходимостью. Структурная единица этого процесса, с психологической точки зрения, — это задача в широком смысле (проблемная, конфликтная ситуация). Задача, как цель, данная в определенных условиях с выделенными известным и искомым. Любое обучение, как и интеллектуальные тренировки, основываются на особенностях мышления и методиках развития интеллекта.

Стандартные требования к составу участников команды по программированию, предъявляемые организаторами олимпиад, зачастую прямо указывают на ролевые функции каждого из членов. В соответствии с ними можно рассмотреть психологические портреты участников команды.

2. УЧЕТ ИНДИВИДУАЛЬНЫХ ОСОБЕННОСТЕЙ МЫШЛЕНИЯ И РАСПРЕДЕЛЕНИЕ РОЛЕЙ В КОМАНДЕ

Участие в интеллектуальных соревнованиях всегда связано с особенностями мышления индивида, его способностью в стрессовой неопределенной ситуации достигнуть поставленных целей. В зависимости от стандартности-нестандартности постановки успешно решаемых задач и методов решения, используемых при этом, можно говорить о следующих видах, а точнее, стилях мышления:

— алгоритмическое мышление (стандартные задачи, решаемые стандартными методами);
— дискурсивное (нестандартные задачи решаются на основании системы взаимосвязанных стандартных умозаключений);
— творческое мышление (принципиально новые решения старых задач, приводящие к открытиям);
— эвристическое мышление (продуктивное мышление, состоящее в решении нестандартных задач).

Также можно говорить о видах мышления, в зависимости от психологических особенностей процесса познания субъектом реальности (интуитивный, аналитический, реалистический, аутистический, эгоцентрический, продуктивный и репродуктивный). Но подготовка к олимпиаде по программированию не предполагает психологической корректировки личности, поэтому работа должна вестись с учетом уже существующих индивидуальных особенностей мышления и процесса познания. Необходимо развивать уже преобладающий стиль мышления.

Заметим, что в данный момент не рассматривается вопрос о развитии команды как естественном процессе, нередко длящегося годами. Идеи командных методов работы заимствованы из мира спорта и стали активно внедряться в практику в конце XX века. Тимбилдинг (от англ. *team building* — построение команды) охватывает широкий диапазон действий для создания и повышения эффективности работы команды.

Организация тренировочного процесса должна исходить из наличия двух групп особенностей индивидуального стиля деятельности (ИСД) человека: ядра и пристройки [1]. Формирование команды на основе психологических особенностей позволит разделить ядро и пристройку индивидуального стиля когнитивной деятельности. Так как ядро ИСД человека основывается на свойствах его нервной системы, не осознается им, то и не поддается коррекции. ИСД содержит особенности, как благоприятствующие, так и препятствующие успеху в командном программировании. Поэтому процесс тренировок должен быть направлен на особенности пристройки ИСД, как на способствующие успеху, так и на компенсирующие некоторые особенности ядра. При этом необходимо постоянно следить за «непрерывностью» возрастания сложности задач и методов их решения, чтобы избежать проблем, связанных с согласованностью между направлением обучения и достигнутым уровнем развития. Тренировки, как и любое обучение, должны быть направлены, согласно Л.С. Выготскому, на «зону ближайшего развития» [2].

Командная олимпиада по программированию — это командное соревнование, которое может проводиться в различных форматах. Наиболее удачным считается формат

ACM ICPC. Команда состоит из трёх участников (может быть еще один запасной участник на случай, когда кто-то не сможет приехать). Тур олимпиады происходит следующим образом: каждой команде выдаётся один компьютер и от шести до тринадцати задач. Время на выполнение заданий–пять часов. Команды пишут решения на языках программирования — чаще всего это C, C++, Python или Java — и посылают их на тестирующий сервер. Программы тестируются на большом количестве различных входных тестов, не известных участникам. Если программа выдала неправильный ответ или не уложилась в ограничения по времени или по памяти, то пославшая её команда получает об этом сообщение и может послать исправленную версию. Задача считается решенной, если программа выдала правильные ответы на всех тестах. В отличие от других олимпиад, частичные решения не учитываются.

Исходя из условий соревнований, выделим роли, которые должны выполнять члены команды. В составе команды должен быть хотя бы один человек, умеющий программировать на одном из языков программирования. Желательно, чтобы в команде был математик-алгоритмист, придумывающий идеи решений и составляющий алгоритм устно или на бумаге (пока компьютер занят). Третьим участником может быть тестировщик. Он придумывает тесты для проверки правильности решения, а также может следить за правильностью написания кода программистом. Для участия в соревнованиях международного уровня хотя бы один человек из команды должен владеть английским языком на уровне перевода условий задач.

Команда любых интеллектуальных соревнований может быть условно разделена на идеологов, исполнителей и критиков. Идеолог–человек, стиль мышления которого позволяет воспринимать любую нестандартную ситуацию (задание), выделять в ней стабильные факторы, на основании которых ставить достижимые цели, систематизировать пути их достижения. Такие люди даже из стандартной конфликтной ситуации пытаются найти нестандартные — «красивые» пути выхода. Задача критика — создавать конфликтную ситуацию на каждом из этапов продвижения к цели. Поиск нестандартных методов — вот «конек» критика. В простонародье про таких людей говорят–«он не ищет легких путей». Исполнитель — «рабочая лошадка» любого проекта. Ему некогда отвлекаться на излишества нестандартности ситуации, понятность задачи является залогом правильности ее достижения, качественной работы. Решая стандартную задачу, исполнитель может использовать и нестандартные методы.

В олимпиадной команде по программированию у каждого должны быть сформированы общекультурные компетенции (ОК) и общие профессиональные компетенции (ОПК) высшего профессионального образования. Сформированности же профессиональных компетенций и дополнительных профессиональных компетенций (ПК, ДПК) достаточно у некоторых участников команды. Соотношение компетенций высшего профессионального образования и психологических особенностей мышления для определения роли участника команды по олимпиадному программированию представлены в таблице 1.

Таким образом, для формирования олимпиадной команды по программированию важно соблюдать принцип учета индивидуальных особенностей процесса мышления и принцип распределения ролей в команде.

3. ЭТАПЫ СИСТЕМЫ ТРЕНИРОВОК ПО КОМАНДНОМУ ПРОГРАММИРОВАНИЮ

Соревновательная деятельность — это особый вид деятельности человека, даже если это интеллектуальная соревновательная деятельность. Как и в спорте, целью психоло-

Таблица 1. Соотношение компетенций высшего профессионального образования и психологических особенностей мышления для определения роли участника команды

Стиль мышления / Сформированные компетенции	**Алгоритмическое**	**Дискурсивное**	**Творческое**	**Эвристическое**
ОК, ОПК	Исполнитель	Исполнитель	Критик	Идеолог
ПК	Исполнитель	Исполнитель Критик	Критик Идеолог	Идеолог
ДПК	Исполнитель	Критик Идеолог	Критик Идеолог	Идеолог

гической подготовки к соревнованиям необходимо рассматривать адаптацию к соревновательным ситуациям, совершенствование и оптимизацию реактивности отражения и ответной реакции на специфичные экстремальные условия, но с учетом особенностей когнитивной деятельности человека. В соревнованиях по командному программированию экстремальная составляющая связана с ограничениями времени, отведенного на решение задач, что существенно влияет на содержание тренировок.

Необходимость структурирования содержания тренировок по командному программированию вызвана широтой и объемностью. Тренировки, наряду с изучением известных методов решения олимпиадных задач, должны включать изучение приемов разработки собственных методов. Одним из важных критериев оценки решения олимпиадных задач является временная сложность представленного решения, поэтому возникает необходимость изучения методов сравнительного анализа времени выполнения различных реализаций. Система тренировок команды по программированию должна включать следующие этапы:

— изучение стандартных методов решения олимпиадных задач;
— разработку собственных методов решения на основе пройденного материала;
— изучение методов оценки различных вариантов решения;
— применение полученных знаний в нестандартных ситуациях.

Для каждого из этапов должны быть подобраны соответствующие серии задач, упорядоченные по возрастанию сложности. Пример такой классификации рассмотрен в [3]: задачи-алгоритмы, задачи-исследования, генерирование стандартных и нестандартных задач. Рассмотрим эти этапы на примере олимпиадных задач, сводящихся к задаче о рюкзаке. Классическую задачу о рюкзаке в общем виде можно сформулировать следующим образом: из заданного множества предметов со свойствами «стоимость» и «вес» требуется выбрать некоторое подмножество предметов таким образом, чтобы получить максимальную суммарную стоимость при соблюдении ограничения на суммарный вес.

Сформулируем задачу о рюкзаке на математическом языке.

Пусть дано N предметов, W — вместимость рюкзака. Обозначим за $w_1, w_2, \ldots, w_N$ — положительные веса соответствующих предметов, а за $c_1, c_2, \ldots, c_N$ — стоимость соответствующих предметов. Нужно найти набор бинарных величин $b_1, b_2, \ldots, b_N$, где $b_i = 1$, если предмет i включен в набор, и $b_i = 0$, если предмет i не включен в набор, и такой что: $b_1 w_1 + \ldots + b_N w_N \leqslant W$; $b_1 c_1 + \ldots + b_N c_N$ — максимальна.

Такая формализация является формулировкой одного из нескольких видов задач о рюкзаке. Данный вид задач принято называть «Рюкзак 0-1». Если предметы нельзя делить на части, то это задача о дискретном рюкзаке, иначе — о непрерывном. Существуют также другие виды задачи о рюкзаке [4].

На первом этапе учащиеся изучают стандартные методы решения задач такого вида, сначала самые простые, затем более сложные. В случае задачи о непрерывном рюкзаке нужно просто отсортировать предметы по стоимости и набирать по максимуму, пока вес набора не достигнет W. Если предметов не больше 20, можно просто перебрать все подмножества и выбрать наиболее подходящее. Оценка сложности переборного алгоритма $O(2^n)$. Если W достаточно мало, чтобы завести массив $[0 \ldots W]$, можно решать ее динамическим программированием за $O(n^W)$.

Приведем пример задачи такого вида с решением. Есть n монет номиналом $c[1], c[2], \ldots, c[n]$. Нужно определить, какое минимальное количество монет требуется, чтобы составить сумму S, или сказать, что это невозможно.

Пусть при составлении суммы мы можем использовать одну монету несколько раз. Введем динамическую переменную $dp[i]$ — сколько нужно монет, чтобы составить сумму i, и, очевидно, что $dp[0] = 0$.

Для всех монет от 1 до n проверяем, выгодно ли нам использовать ее при составлении суммы i: $dp[i] = min(dp[i], dp[i - c[j]] + 1), j = 1 \ldots n$.

Исходя из формулировки задачи, очевидны следующие требования: $c[j]$ должно быть не больше i, и сумма $i - c[j]$ должна быть составима. Рассмотрим вариант решения, записанный на одном из языков программирования с комментариями:

```
int n,s;
int c[105];
int dp[10005];

int main(){

        scanf("%d",&n);
        scanf("%d",&s);

        For(i,1,n)scanf("%d",&c[i]);

        Fill(dp,-1); //сначала все суммы несоставимы
        dp[0]=0; //чтобы получить сумму 0 монеты не нужны=)

        For(i,1,s)
        For(j,1,n)
        if(i-c[j]>=0\&\&(dp[i-c[j]]!=-1)){ //номинал монеты не больше
                  //необходимой суммы, и есть способ составить остаток
                 if(dp[i]==-1)
                 dp[i]=dp[i-c[j]]+1; //если еще не был найден способ
                                     //составить сумму i
                 else
                 dp[i]=min(dp[i],dp[i-c[j]]+1); //если был, пробуем улучшить
        }
        if(dp[s]==-1)
        printf("Impossible\n");
        else
        printf("%d\n",dp[s]);

        return 0;
}
```

Рассмотрим также случай, когда мы не можем использовать одну монету больше одного раза. Принцип динамики не меняется, но нам придется для каждой монеты, которую мы пытаемся использовать, составлять суммы, начиная с самой большой.

Еще один метод решения рассматриваемой задачи — метод «meet-in-the-middle». Он работает в тех случаях, когда предыдущие методы неприменимы, и состоит из следующих шагов. Разобьем предметы на две примерно равные части. Переберем для первой половины все подмножества, для каждого из них будем записывать его вес в массив $a[\,]$. Аналогично поступим со второй половиной, но вес будем записывать в массив $b[\,]$.Теперь отсортируем массив $a[\,]$ по возрастанию, и для каждого элемента из $b[\,]$ будем определять, сколько элементов в массиве $a[\,]$ не превосходят $c-b[i]$, то есть со сколькими элементами $a[\,]$ мы можем соединить текущее подмножетсво $b[i]$, чтобы не превысить максимальный вес. Это можно определить бинарным поиском.

Оценка сложности этого алгоритма $O(2^{(n/2)})$. Далее предложим учащимся самостоятельно найти решение для нескольких задач, сводящихся к задаче о рюкзаке [5–7].

На втором этапе рассмотрим частный случай задачи о неограниченном рюкзаке. Задача о неограниченном рюкзаке — обобщение классической задачи, когда любой предмет может быть взят любое количество раз. В данном частном случае будем считать, что стоимость предмета равна весу предмета, поэтому в дальнейшем стоимость предметов можно не учитывать. Также добавим условие, что обязательно нужно взять ровно M предметов. Заранее можно договориться, что все веса предметов различны между собой. Если же есть какие-то предметы с одинаковым весом, то будем всегда брать из них только какой-то определенный. Такое допущение ничего не изменит в задаче, так как любой предмет можно брать любое количество раз.

Приведём следующий алгоритм решения данной задачи, предложенный одним из участников команды, который существенно отличается от рассмотренных ранее. По множеству весов всех предметов определить многочлен y как $y = \{x^{w_i} \mid 1 \leqslant i \leqslant N\} = x^{w_1} + \ldots + x^{w_N}$. Возвести многочлен в степень M. Это можно сделать обычным «наивным» способом. Мы рассмотрели несколько более эффективных вариантов: с использованием двоичной лунной арифметики, применяя быстрое преобразование Фурье и распараллеленное быстрое преобразование Фурье. Более подробно реализация перечисленных способов описана в статье [8]. Ответом на задачу является $R = max\{i \mid i \leqslant W \wedge y_i^M \neq 0\}$.

При переходе на третий этап тренировочного процесса можно рассматривать различные критерии оценивания решений олимпиадных задач. Одним из важных критериев является временная сложность представленного решения, которую можно оценить с помощью тестирования. Для тестирования выбирают варьируемые параметры входных данных рассматриваемого алгоритма. Например, нами был проведен сравнительный анализ времени выполнения программ среди различных решений задачи. Тестирование было проведено на случайных многочленах степени N, которые требуется возвести в M-ю степень. Приведём ограничения на N и M в тестах:

тест 1–$N = 500, M = 500$;
тест 2–$N = 500, M = 1000$;
тест 3–$N = 1000, M = 500$;
тест 4–$N = 1000, M = 1000$.

Рассмотрено пять различных решений задачи:

— решение A–решение с помощью динамического программирования;

— решение *B*–решение с помощью динамического программирования, используя битовые маски;
— решение *C*–решение с помощью произведения многочленов в лунной арифметике (произведение реализовано наивным образом);
— решение *D*–решение с помощью произведения многочленов в лунной арифметике (произведение многочленов реализовано с помощью быстрого преобразования Фурье);
— решение *E*–решение с помощью произведения многочленов в лунной арифметике (произведение многочленов реализовано с помощью распараллеленного быстрого преобразования Фурье).

Тестирование проводилось на ноутбуке с процессором AMD Phenom™ II P840 TripleCoreProcessor 1.90 GHz.

Результаты времени выполнения программ представлены в таблице 2. Решение задачи с помощью распараллеленного быстрого преобразования Фурье в лунной арифметике является более быстрым по сравнению с остальными рассмотренными нами решениями.

Таблица 2. Результаты тестирования

Тест	Решение A, с.	Решение B, с.	Решение C, с.	Решение D, с.	Решение E, с.
1	202.051	2.035	13.866	0.759	0.607
2	778.803	8.119	55.649	1.648	1.242
3	813.130	6.559	61.022	1.645	1.263
4	3009.574	30.588	276.088	3.456	2.632

Заключительный этап подготовки может быть связан с рассмотрением применения в различных областях знаний. Некоторые применения задачи о рюкзаке для подсчета числа помехоустойчивых циклических кодов рассматривались в статьях [9, 10].

4. ЗАКЛЮЧЕНИЕ

Формирование команды по программированию необходимо основывать на индивидуальных особенностях мышления каждого из ее членов. Четкое распределение ролей позволит наиболее полно использовать методики развития интеллекта, адаптированные к соответствующему тренировочному процессу. Четкая систематизация задач, направленная на опережающее развитие интеллекта, позволяет корректировать пристройку индивидуального стиля когнитивной деятельности. Полученные авторами результаты подтверждаются практически. Разработанная система тренировок и принципы формирования команды позволили пройти внутренние отборочные этапы и достойно выступить в чемпионате мира по командному программированию.

Список литературы

1. *Климов Е. А.* Индивидуальный стиль деятельности в зависимости от типологических свойств нервной системы: к психологическим основам научной организации труда, учения, спорта. Казань: Издательство Казанского университета, 1969.
2. *Выготский Л. С.* Мышление и речь. М., 1999.

3. *Толкачева Е. А., Казакевич В. Г.* Классификация задач на основе методов познания // Труды II Междунар. научно-практической конф., посвященной 125-летию П. А. Ларичева «Задачи в обучении математике, физике, информатике: теория, опыт, инновации». Вологда: ИП Киселев А. В., 2017. С. 151–155.
4. *Martelo, S., Toth, P.* Knapsack problems. NY: Wiley, 1990. 306 с.
5. Куча камней. URL: http://acm.timus.ru/problem.aspx?space=1&num=1005 (дата обращения 14.06.18).
6. Квадратная страна. URL: http://acm.timus.ru/problem.aspx?space=1&num=1073 (дата обращения 14.06.18).
7. Джентельмены. URL: http://acm.timus.ru/problem.aspx?space=1&num=1244 (дата обращения 14.06.18).
8. *Чесноков А. И.* Об алгоритме решения задачи о рюкзаке на основе лунной двоичной арифметики // Материалы 5-й научно-практической internet-конференции «Междисциплинарные исследования в области математического моделирования и информатики», Ульяновск, SIMJET, 2015. С. 52–56.
9. *Корабельщикова С. Ю., Чесноков А. И.* О числе различных циклических кодов заданной длины // Вектор науки ТГУ. 2013. № 4(26). С. 25–26.
10. *Зяблицева Л. В., Корабельщикова С. Ю., Чесноков А. И.* Линейные коды, исправляющие ошибки, и алгоритмы их подсчёта // Эвристические алгоритмы и распределённые вычисления, Т. 1, вып. 3, 2014. С. 47–59.

Поступила в редакцию 27.09.2018, окончательный вариант — 01.11.2018.

Computer tools in education, 2018
№ 6: 47–55
http://ipo.spb.ru/journal
doi:10.32603/2071-2340-2018-6-47-55

PRINCIPLES OF TEAM-FORMATION FOR A PROGRAMMING CONTEST

Korabelshchikova S. Y.[1], Tolkacheva E. A.[2], Butin K. P.[1]

[1]Northern (Arctic) Federal University named after M. V. Lomonosov, Arkhangelsk, Russia
[2]Saint-Petersburg Electrotechnical University ETU "LETI, Saint-Petersburg, Russia

Abstract

The issue of organizing and training a team for participating in collegiate programming contests has been The issue of organizing and training a team for participating in collegiate programming contests has been considered. The principles of team-formation, along with the role of each member, based on psychological intellection features and competency level, have been developed. The classification of intellection types was chosen based on typical or non-typical problems and operational intellection procedures. Often in programming contests there are problems that can be reduced to the knapsack problem. Given in the paper is a system of exercises that can be reduced to the knapsack problem. The exercises have been arranged in order of ascending complexity. First, the unlimited knapsack problem and different algorithms for solving it are considered. For the special case of this task, when the value of the item is equal to its weight, the algorithm and its various implementations are presented. The developed training system and the team-formation principles allowed to achieve great result, in particular, to go

through the internal qualifying stages and adequately perform in the International Collegiate Programming Contest, which confirms the effectiveness of the proposed methods.

Keywords: *programming contest, team contest, intellection types, training system, higher professional education competencies, knapsack problem.*

Citation: S. Y. Korabelshchikova, E. A. Tolkacheva, and K. P. Butin, “Principles of Teamformation for a Programming Contest”, *Computer tools in education*, no. 6, pp. 47–55, 2018 (in Russian). doi:10.32603/2071-2340-2018-6-47-55

Received 27.09.2018, the final version — 01.11.2018.

Svetlana Y. Korabelshchikova, associate professor at Department of Information Technologies and Security, Northern (Arctic) Federal University named after M. V. Lomonosov, s.korabelsschikova@narfu.ru
Elena A. Tolkacheva, associate professor at the Department of Algorithmic Mathematics, Saint-Petersburg Electrotechnical University ETU "LETI" ; 197376 Saint-Petersburg, ul. Professora Popova 5, korp. 3, Dep. of Algorithmic Mathematics, eatolkacheva@etu.ru
Kirill P. Butin, assistant professor at Department of Information Technologies and Security, Northern (Arctic) Federal University named after M. V. Lomonosov, k.butin@narfu.ru

**Корабельщикова Светлана Юрьевна,
доцент кафедры информатики
и информационной безопасности
САФУ им. М. В. Ломоносова,
s.korabelsschikova@narfu.ru**

**Толкачева Елена Алексеевна,
кандидат физико-математических наук,
доцент, заместитель заведующего
кафедрой Алгоритмической математики
СПбГЭТУ «ЛЭТИ» им. В.И. Ульянова
(Ленина); 197376, Санкт-Петербург,
ул. Профессора Попова, д. 5, корп. 3,
кафедра алгоритмической математики,
eatolkacheva@etu.ru**

**Бутин Кирилл Павлович,
ассистент кафедры информатики
и информационной безопасности
САФУ им. М. В. Ломоносова,
k.butin@narfu.ru**