where $1-\frac{2}{n} < m_1 \le ... \le m_{i_o} < 1 < m_{i_o+1} \le ... \le \mathrm{m}_n < m + \frac{2}{n}$, we prove that the singularity of solution $u(\mathrm{x},\mathrm{t})$ at the point $(x_o, 0)$ is removable.

We formulate the removability result in the form of behavior of the function

$$M(\mathrm{r}) = ess\ \sup\{|\,\mathrm{u}(\mathrm{x},\mathrm{t})\,| : (\mathrm{x},\mathrm{t}) \in \mathrm{D}(\mathrm{R}_o) \setminus \mathrm{D}(\mathrm{r})\},$$

where

$$D(\mathrm{r}) = \left\{ (\mathrm{x},\mathrm{t}) \in \Omega_T : \sum_{i=1}^{n} \left(\frac{|\,\mathrm{x}_i - \mathrm{x}_i^{(0)}\,|}{r^{k_i}} \right)^2 + \frac{t}{r^k} \le \right\}, \mathrm{k} = \mathrm{n}(\mathrm{m}-1) + 2,\ \ \mathrm{k} = \frac{\mathrm{n}(\mathrm{m}-m_i)+2}{2}.$$

Our main result is the following theorem.

Theorem 1

Assume that condition (3) is fulfilled. Let u be a weak solution of the problem (1), (2) satisfying equality

$$\lim_{r \to 0} M(\mathrm{r})\,\mathrm{r}^n = 0,$$

then the singularity at the point $(x_o, 0)$ is removable.

References

1. Kolodij I. M., On boundedness of generalized solutions of parabolic differential equations, Vestnik Moskov. Gos. Univ. 5, pp. 25-31 (1971)
2. Namlyeyeva Yu. V., Shishkov A. E., Skrypnik I. I. Removable isolated singularities for solutions of doubly nonlinear anisotropic parabolic equations, Applicable Analysis, 89(10), 1559–1574 (2010)

Підсекція прикладної математики та комп'ютерних наук

УДК 512.548.7

ЗАСТОСУВАННЯ ЛАТИНСЬКИХ КВАДРАТІВ ПРИ СКЛАДАННІ РОЗКЛАДУ ТУРНІРУ

А. С. Акопян, Ф. М. Сохацький

Для того щоб провести турнір з гольфу необхідно скласти розклад турніру, який має задовольняти умовам:

1) 15 гравців в гольф грають в 5 трійках (таким чином, що кожен гравець грає щотижня);
2) 5 різних стартових майданчиків;
3) ліга триває протягом 5 тижнів;
4) жодні двоє гравців в гольф не можуть бути в трійці більше ніж один раз;
5) жоден гравець не починає гру двічі на одному стартовому майданчику.

Ця задача зводиться до побудови трійки попарно ортогональних латинських квадратів. Нагадаємо, що квадрат заповнений елементами множини Q називається *латинським*, якщо кожен рядок та кожен стовпець цього квадрату є перестановкою множини Q . Кожній операції на множині Q відповідає квадрат, а квазігрупі – латинський квадрат, який є внутрішньою частиною її таблиці Келі.

Нехай квадрат L є внутрішньою частиною таблиці Келі операції $(*)$, яка визначена на множині Q. Він є латинським тоді і тільки тоді, коли $(Q;*)$ – квазігрупа, тобто для будь-яких $a,b \in Q$ існують єдиний x та єдиний y такі що

$$a * x = b \quad \wedge \quad y * a = b.$$

Наприклад, оскільки 2 та 1 є оборотними у $(\mathbb{Z}_5, +, \cdot)$, то операції $x * y = 2x + y$ над $(\mathbb{Z}_5, +, \cdot)$ відповідає латинський квадрат

0	1	2	3	4
2	3	4	0	1
4	0	1	2	3
1	2	3	4	0
3	4	0	1	2

Два квадрати називаються ортогональними, якщо при їх накладанні всі утворені пари є різними.

Завдання складання графіку турніру з гольфу було вирішено шляхом використання трьох попарно ортогональних латинських квадратів $(\mathbb{Z}_5; \cdot)$, $(\mathbb{Z}_5; °)$, $(\mathbb{Z}_5; *)$, де $\mathbb{Z}_5$ – кільце лишків за модулем 5, що складається з класів лишків $\{0, 1, 2, 3, 4\}$. В кожному з наведених нижче латинських квадратів складено розклад для одного гравця з команд 0, 1, 2, 3, 4. Тобто, першого тижня перший гравець команди 0 грає на майданчику 0, перший гравець команди 3 грає на майданчику 1 тощо.

Використання латинських квадратів дозволяє виконати умови 2 (п’ять стовпчиків кожного квадрату – це п’ять різних стартових майданчиків), 3 (п’ять рядків кожного квадрату – це п’ять тижнів, які триватиме турнір) та 5 (жоден гравець кожної з команд не починає гру двічі на одному стартовому майданчику). Враховуючи те, що ці латинські квадрати попарно ортогональні, усі 15 гравців грають щотижня в рівних умовах у п’яти трійках, тобто виконується умова 1, і жоден з гравців незалежно від команди не грає двічі з іншим гравцем в трійках більше ніж одного разу, тобто виконується умова 4.

·	**0**	**1**	**2**	**3**	**4**
0	0	3	1	4	2
1	4	2	0	3	1
2	3	1	4	2	0
3	2	0	3	1	4
4	1	4	2	0	3

°	**0**	**1**	**2**	**3**	**4**
0	0	1	2	3	4
1	2	3	4	0	1
2	4	0	1	2	3
3	1	2	3	4	0
4	3	4	0	1	2

*	**0**	**1**	**2**	**3**	**4**
0	0	2	4	1	3
1	3	0	2	4	1
2	1	3	0	2	4
3	4	1	3	0	2
4	2	4	1	3	0

Накладаємо ці три латинські квадрати і отримаємо масив S

000	312	124	431	243
423	230	042	304	111
341	103	410	222	034
214	021	333	140	402
132	444	201	013	320

в якому кожна клітинка S містить трійку (i, j, k), де i береться з $(\mathbb{Z}_5; \cdot)$, j з $(\mathbb{Z}_5; °)$, а k – з $(\mathbb{Z}_5; *)$. 15 гравців складаються з 5 символів з $(\mathbb{Z}_5; \cdot)$, плюс 5 символів з $(\mathbb{Z}_5; °)$, плюс 5 символів з $(\mathbb{Z}_5; *)$. Тижні турніру будуть рядками масиву S, а стартові майданчики – стовпчиками.

Було отримано п'ять тижнів турніру для 15 гравців в гольф. Наприклад, розглянемо перший тиждень турніру. Враховуючи те, що тижні турніру – рядки масиву S, а стартові майданчики – стовпчики, то на першому тижні на першому стартовому майданчику зіграють троє гравців з команди 0. На другому стартовому майданчику зіграє перший гравець з команди 3, перший гравець з команди 1 та перший гравець з команди 2. На

третьому стартовому майданчику зіграє другий гравець з команди 1, другий гравець з команди 2 та перший гравець з команди 4. На четвертому стартовому майданчику зіграє другий гравець з команди 4, другий гравець з команди 3 та третій гравець з команди 1. На п'ятому стартовому майданчику зіграє третій гравець з команди 2, третій гравець з команди 4 та третій гравець з команди 3. Таким чином, ми отримали розклад турніру.

Література

1. Белоусов В. Д. Латинские квадраты, квазигруппы и их приложения. / В. Д. Белоусов, Г. Б. Белявская. – Кишинев: Штиница, 1989. – 78 с.

УДК 004:371.8:378

ПРОБЛЕМА ОБРАННЯ МОВИ ПРОГРАМУВАННЯ, ЯК ІНСТРУМЕНТУ ДЛЯ НАВЧАННЯ ТА РОЗРОБКИ

Ю. С. Антонов, К. Р. Дзігора

Питання, яку мову програмування обрати та яка краща, дуже часто виникає як у тих, хто збирається вивчати програмування(вже вивчає), так і у тих, хто ці мови програмування викладає [1, 2]. Зрозуміло, що кожна сторона має свої критерії обрання тієї чи іншої мови програмування. Існують роботи [2], де пропонується обирати мову програмування в залежності від задачі та її предметної галузі, вимог до процесу розробки, швидкодії та потреби у ресурсах, тощо. Мета цієї роботи висвітлити деякі проблеми обрання мови програмування та показати, що кожна мова програмування є інструментом для розв'язання певного класу задач. Автори, у жодному випадку, не ставлятьза метудовести перевагу використання якоїсь мови програмування.

Бажаючих вивчати програмування можна поділити на дві категорії: перша – не знає жодної мови програмування, друга - знає хоча б одну мову програмування. Ті, хто вивчає вже не першу мову програмування, роблять це для кар'єрного зростання, самовдосконалення, підвищення конкурентоспроможності, тощо. Ті, хто не знає жодної мови програмування, фактично схожі на чисту дошку, на якій буде "писати" той чи інший наставник, і віднього залежить дуже багато. Серед усіх проблем, що виникають під час навчання, хотілося б виділити наступні: відсутність мотивації до навчання, недостатність знань (досвіду) та «євангелізм». Відсутність мотивації це проблема людей з першої категорії, напрям навчання яких часто обрано іншими людьми. Друга проблема стосується як викладачів так і слухачів: недостатньо знань або вмінь для опанування курсу (розв'язання задачі), вміння писати програми, але відсутність навичок доступно пояснити, як це робити, тощо.Найбільш загрозливою проблемою у наш час можна вважати «євангелістів» тієї чи іншої мови програмування(IT технології) та подібних до них людей. Такі люди, свідомо чи несвідомо, формують світогляди або стереотипи навколо тієї чи іншої технології. Лаври «універсальної»або «самої крутої» мови програмування вже отримували C та C++,Java таC#, PHP таPython. Якщо в один той самий час, дві компанії будуть проводити навчання або інші заходи, то ми можемо почути в одному таборі що «*мова X краща за мову Y*», а в іншому – навпаки *«Y значно крутіша ніж X*». Не кожна людина, яка відвідає обидва заходи зможе зробити правильні висновки. Наприклад, для задач з однієї предметної галузі підходить мова X, а з другої – Y, а для третьої – Z. У математичному аналізі одна й та сама теорема може мати два і більше доказів, під час розгляду яких кажуть, що, наприклад, перший доказ є коротким, але складним, а другий – легше сприймається, але вдвічі довший. Так само, при порівнянні алгоритмів, ми кажемо, що перший – найшвидший, а другий використовує