

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
**АДЫГЕЙСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**  
Инженерно-физический факультет  
Кафедра автоматизированных систем обработки информации и  
управления

ОТЧЕТ ПО ПРАКТИКЕ

*Написать программу генератор случайных чисел  
Парка-Миллера с перетасовкой и без  
перетасовки, вариант 2.*

2 курс, группа 2ИВТ1

Выполнили:

\_\_\_\_\_ Токмаков И.М.

«\_\_\_» \_\_\_\_\_ 2022 г.

Руководитель:

\_\_\_\_\_ С. В. Теплоухов

«\_\_\_» \_\_\_\_\_ 2022 г.

Майкоп, 2022 г.

# 1. Введение

- 1) Текстовая формулировка задачи
- 2) код данной задачи
- 3) скриншот программы

## 2. Вариант 2

задание

Генератор случайных чисел Парка-Миллера с перетасовкой и без перетасовки.

Теория

Самая простая последовательность, которая можно предложить для реализации генератора равномерного распределения:

$$I(j+1) = a * I(j) \pmod{m}$$

при соответствующей выборке констант. Константы были предложены Park и Miller:

$$a = 7^5 = 16807, \quad m = 2^{31} - 1 = 2147483647.$$

Модуль разлагается в выражение:

$$m = a * q + r$$

Если  $r < q$  и  $0 < z < m - 1$ , то при этом величины  $a * (z \bmod q)$  и  $r * [z/q]$  всегда лежат в интервале  $0, \dots, m - 1$ . Для умножения  $(a * z) \pmod{m}$  при этом используется алгоритм:

- $t = a(z \bmod q) - r[z/q]$

- если  $t < 0$ , то  $t += m$ .

- $(a * z) \pmod{m} = t$

В случае констант Парка-Миллера можно использовать  $q = 12773$  и  $r = 2836$ .

## 3. Ход работы

### 3.1. Код приложения

Генератор случайных чисел Парка-Миллера с перетасовкой

```
#include <stdio.h>
```

```
#define IA 16807
```

```
#define IM 2147483647
```

```
#define AM (1./IM)
```

```
#define IQ 12773
```

```
#define IR 2836
```

```
#define NTAB 32
```

```

#define NWUP 8
#define NDIV (1+(IM-1)/NTAB)
#define EPS 1.2e-7
#define RNMX (1.0-EPS)
#define MASK 123456789

static long dummy;
void Seed(long dum) {
    dummy = dum;
}

float unirand0(void) {
    long k;
    float ans;

    dummy ^= MASK;
    k = dummy / IQ;

    if ((dummy = IA * (dummy - k * IQ) - IR * k) < 0)
        dummy += IM;

    ans = AM * dummy;

    dummy ^= MASK;

    return(ans);
}

float unirand1(void) {
    int j;
    long k;
    static long iy = 0, iv[NTAB];
    float temp;

    if (dummy <= 0 || !iy) {
        if (dummy < 0) dummy = -dummy; else
            if (dummy == 0) dummy = 1;
        for (j = NTAB + NWUP - 1; j >= 0; j--) {
            k = dummy / IQ;

            if ((dummy = IA * (dummy - k * IQ) - IR * k) < 0)
                dummy += IM;

            if (j < NTAB)

```

```

iv[j] = dummy;
}

iy = iv[0];
}

k = dummy / IQ;
if ((dummy = IA * (dummy - k * IQ) - IR * k) < 0)
dummy += IM;

iy = iv[j = iy / NDIV];
iv[j] = dummy;

if ((temp = AM * iy) > RNMX)
return(RNMX);
else
return(temp);
}

int main() {
int i;
Seed(6723);
for (i = 0; i < 100; i++)
printf("%f\n", unirand1());
}

```

Генератор случайных чисел Парка-Миллера без перетасовки

```

#include <stdio.h>
#define IA 16807
#define IM 2147483647
#define AM (1./IM)
#define IQ 12773
#define IR 2836
#define MASK 123456789

```

```

static long dummy;
void Seed(long dum) {
dummy = dum;
}

```

```

float unirand0(void) {
long k;
float ans;

```

```

dummy ^= MASK;
k = dummy / IQ;

if ((dummy = IA * (dummy - k * IQ) - IR * k) < 0)
dummy += IM;

ans = AM * dummy;

dummy ^= MASK;

return(ans);
}

int main() {
int i;
Seed(6723);
for (i = 0; i < 100; i++)
printf("%f\n", unirand0());
}

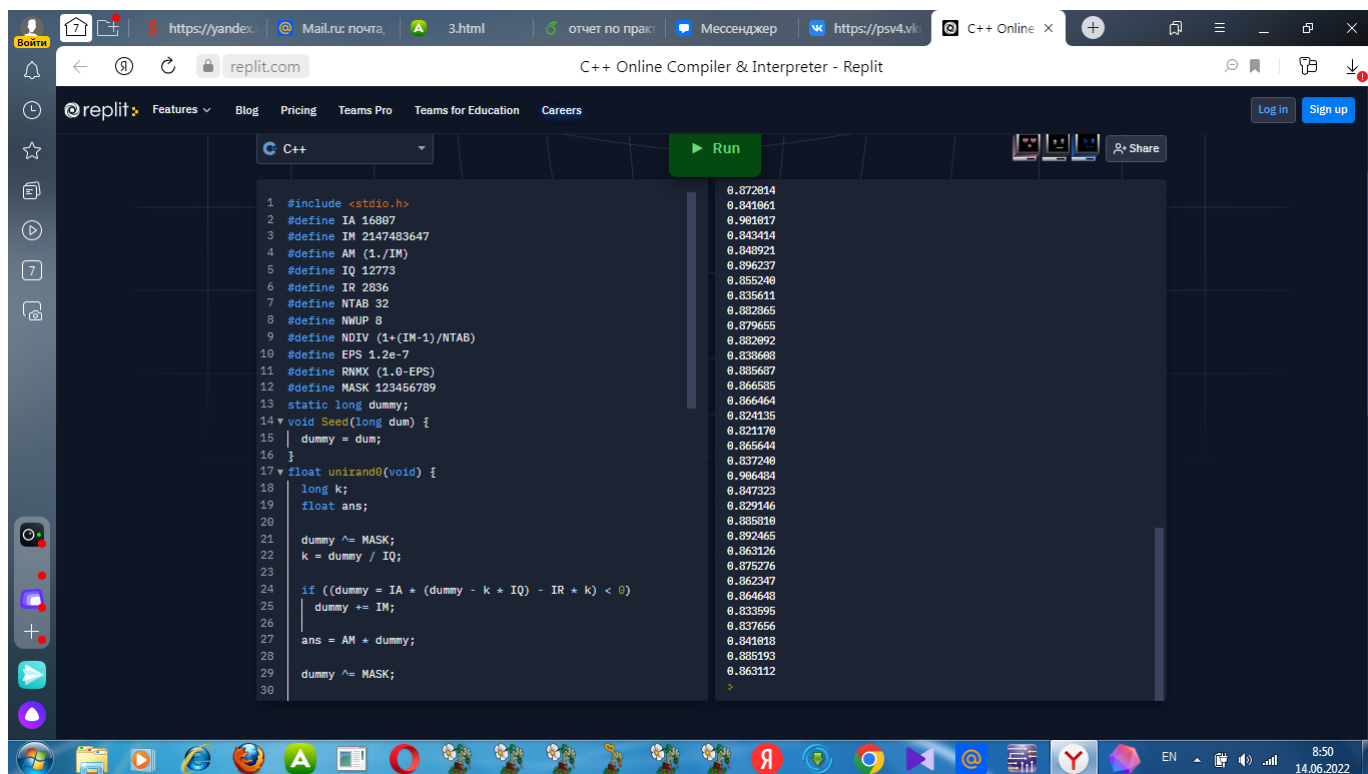
```

### 3.2. формулы

Общая формула генератора случайных чисел  $X_{k+1} = a * X_k \bmod m$

## 4. Пример скриншота программы

Генератор случайных чисел Парка-Миллера с перетасовкой



The screenshot shows a web browser window with the Replit online compiler. The C++ code defines constants for the Park-Miller algorithm and generates a sequence of random numbers. The output is displayed on the right side of the editor.

```
1 #include <stdio.h>
2 #define IA 16807
3 #define IM 2147483647
4 #define AM (1./IM)
5 #define IQ 12773
6 #define IR 2836
7 #define NTAB 32
8 #define NIUP 8
9 #define NDIV (1+(IM-1)/NTAB)
10 #define EPS 1.2e-7
11 #define RNMX (1.0-EPS)
12 #define MASK 123456789
13 static long dummy;
14 void Seed(long dum) {
15     dummy = dum;
16 }
17 float unrand0(void) {
18     long k;
19     float ans;
20
21     dummy ^= MASK;
22     k = dummy / IQ;
23
24     if ((dummy = IA + (dummy - k * IQ) - IR * k) < 0)
25         dummy += IM;
26     ans = AM * dummy;
27     dummy ^= MASK;
28 }
29
```

Output:

```
0.872814
0.841861
0.981817
0.843414
0.848921
0.896237
0.855248
0.835611
0.882865
0.879655
0.882892
0.838688
0.885687
0.866585
0.866464
0.824135
0.821178
0.865644
0.837248
0.986484
0.847323
0.829146
0.885818
0.892465
0.863126
0.875276
0.862347
0.864648
0.833595
0.837656
0.841818
0.885193
0.863112
```

Рис. 1. скриншот программы

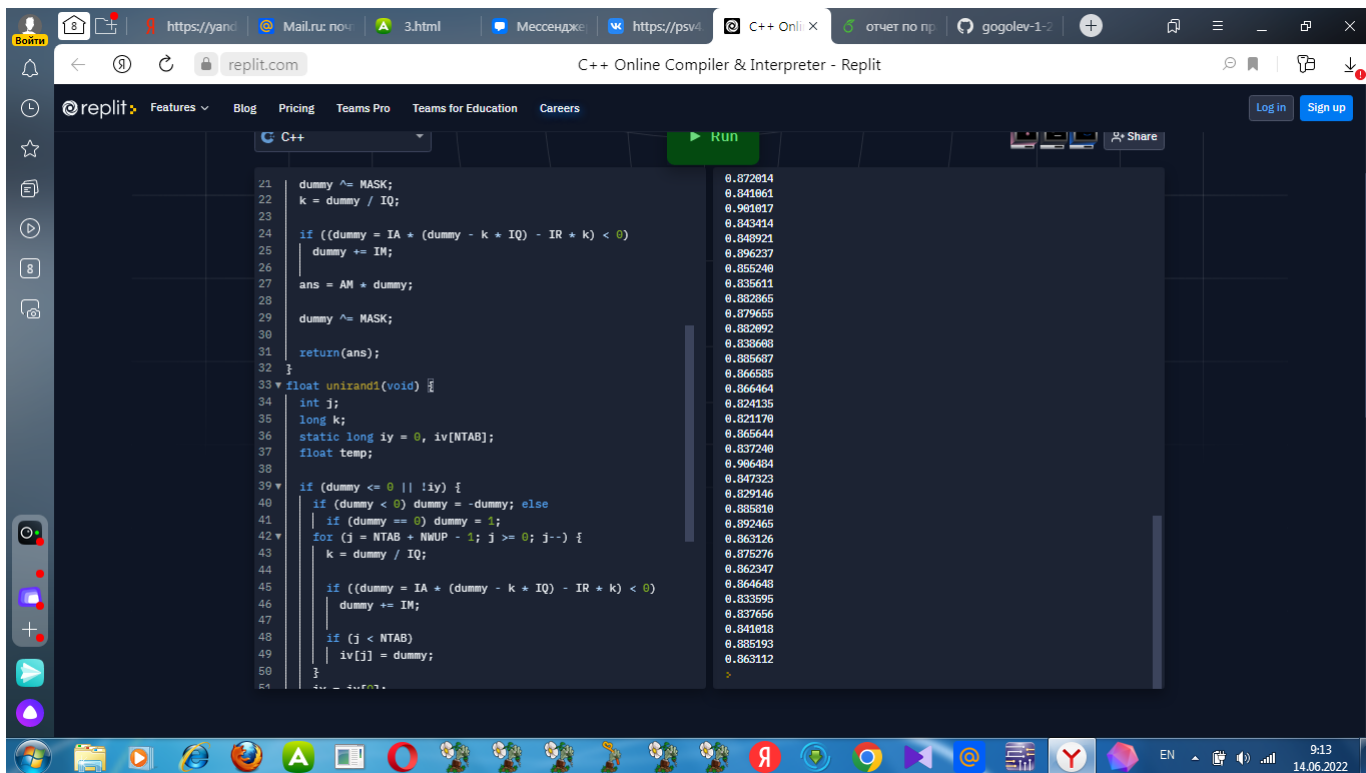


Рис. 2. скриншот программы

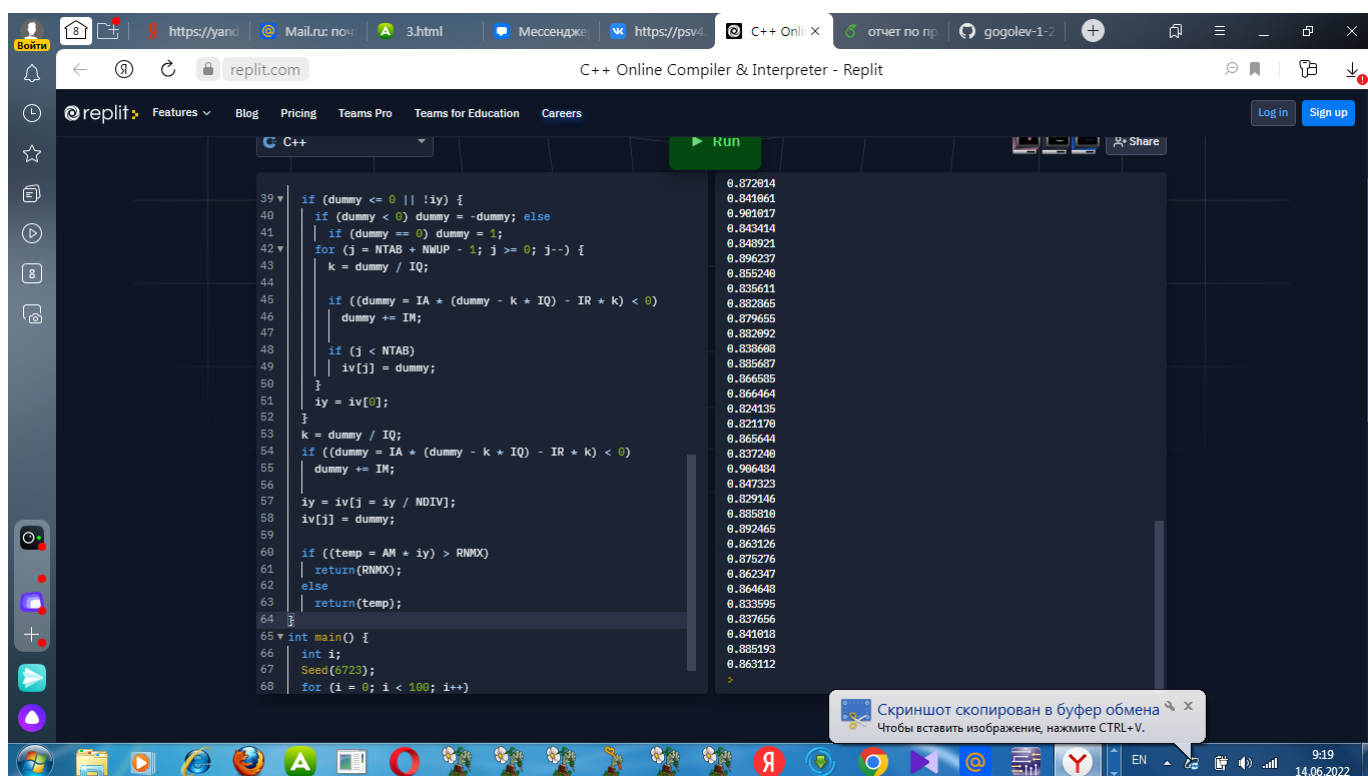


Рис. 3. скриншот программы



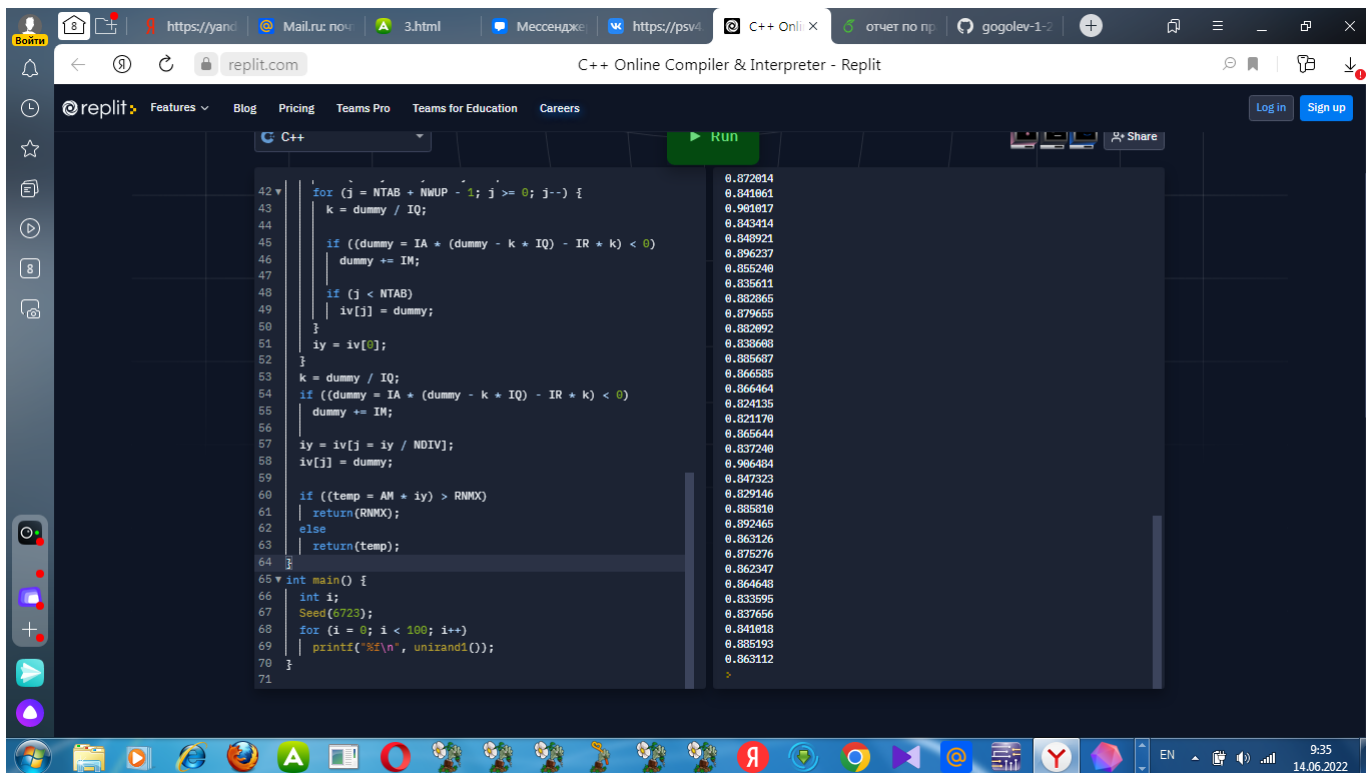
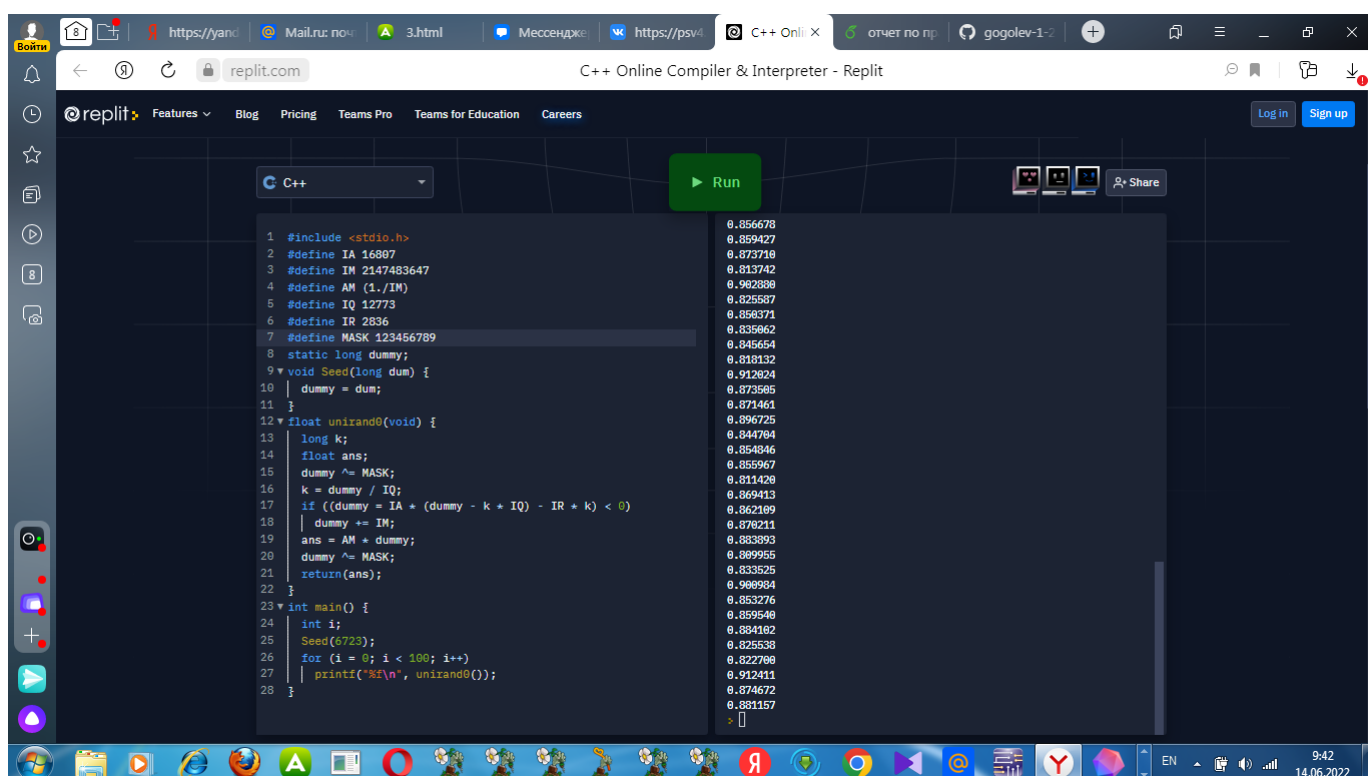


Рис. 4. скриншот программы

## Генератор случайных чисел Парка-Миллера без перетасовкой



The screenshot shows a web browser window with the URL `replit.com`. The page title is "C++ Online Compiler & Interpreter - Replit". The code editor displays a C++ program for generating random numbers using the Park-Miller algorithm. The code includes constants for IA, IM, AM, IQ, IR, and MASK, and functions for seeding and generating random numbers. The output window on the right shows a list of 20 generated random numbers.

```
1 #include <stdio.h>
2 #define IA 16807
3 #define IM 2147483647
4 #define AM (1./IM)
5 #define IQ 12773
6 #define IR 2836
7 #define MASK 123456789
8 static long dummy;
9 void Seed(long dum) {
10     dummy = dum;
11 }
12 float unrand0(void) {
13     long k;
14     float ans;
15     dummy ^= MASK;
16     k = dummy / IQ;
17     if ((dummy = IA * (dummy - k + IQ) - IR * k) < 0)
18         dummy += IM;
19     ans = AM * dummy;
20     dummy ^= MASK;
21     return(ans);
22 }
23 int main() {
24     int i;
25     Seed(6723);
26     for (i = 0; i < 100; i++)
27         printf("%f\n", unrand0());
28 }
```

Output:

```
0.856678
0.859427
0.873718
0.813742
0.962898
0.825537
0.869371
0.835962
0.845654
0.818132
0.912624
0.873585
0.871461
0.896725
0.844704
0.854846
0.855967
0.811428
0.869413
0.862109
0.870211
0.833093
0.899955
0.833525
0.986984
0.853276
0.895408
0.894102
0.825538
0.822708
0.912411
0.874672
0.881157
```

Рис. 5. скриншот программы

## 5. библиографические ссылки

Для изучения «внутренностей»  $\text{\TeX}$  необходимо изучить [1], а для использования  $\text{\LaTeX}$  лучше почитать [2, 3].

## Список литературы

- [1] Кнут Д.Э. Всё про  $\text{\TeX}$ . — Москва: Изд. Вильямс, 2003 г. 550 с.
- [2] Львовский С.М. Набор и верстка в системе  $\text{\LaTeX}$ . — 3-е издание, исправленное и дополненное, 2003 г.
- [3] Воронцов К.В.  $\text{\LaTeX}$  в примерах. 2005 г.