



КАФЕДРА Системы автоматизированного проектирования (РК-6)

по дисциплине: «Разработка программных систем»

## Вариант лабораторной работы 2

## Оценка

*Москва, 2022 г.*

## ОГЛАВЛЕНИЕ

Текст задания .....	3
Описание структуры программы.....	3
Используемые данные .....	3
Блок-схема.....	4
Пример работы программы.....	5
Текст программы.....	5

### **Текст задания**

Составить программу, решающую задачу разрешимости логического выражения, содержащего 3 переменные (т.е. задачу отыскания значений всех комбинаций 3 логических переменных, делающих выражение истинным).

Замечание. Логическое выражение реализовать в виде функции языка СИ с тремя аргументами.

Для проверки истинности логического выражения на каждой комбинации переменных организовать 8 параллельно выполняемых процессов, при этом родственные связи процессов имеют вид дерева с тремя ярусами. В качестве значений логических переменных целесообразно использовать значения, возвращаемые системным вызовом `fork`. В каждом процессе осуществить печать значений переменных и истинности выражения.

### **Описание структуры программы**

Задача реализуется при помощи последовательных вызовов `fork()`. Каждый вызов `fork` порождает новый процесс и возвращает `pid` процесса родителю и 0 себе. `Pid` процесса – это некоторое ненулевое число. Применяя двойное отрицание к возвращаемому функцией `fork()` значению, первое отрицание сделает из этого числа ноль, а второе сделает из нуля один – значения логических переменных (переменная может быть либо 0 либо 1, так как двоичная система счисления). Затем они обрабатываются в функции вместе с введенными логическими операциями.

### **Используемые данные**

Константная переменная `n` задает количество логических переменных в получаемом на вход выражении. В задании необходимо составить таблицу истинности для 3 логических переменных.

Функции `void check(bool *pid, int size, char* buf)` проверяет является ли выражение истиной. Здесь `pid` – массив логических переменных, `size` – размер этого массива, а `buf` – массив логических операций.

Функции `bool switch_op(bool a, bool b, char op)` на вход получает 2 логические переменные и логический оператор и возвращает результат применения оператора к переменным.

### Блок-схема



Рис. 1. Блок-схема реализованного процесса

## Пример работы программы

a*b+c	a+b*c	a+b+c
1 1 1 *	1 1 1 *	1 1 1 *
0 1 1 *	1 1 0 *	0 1 1 *
0 0 1 *	1 0 1 *	1 0 1 *
0 0 0	1 0 0 *	0 0 1 *
0 1 0	0 1 1 *	0 1 0 *
1 0 0	0 1 0	0 0 0
1 1 0 *	0 0 1	1 1 0 *
1 0 1 *	0 0 0	1 0 0 *

Рис. 2. Примеры работы программы для трёх различных входных значений  
a\*b+c, a+b\*c и a+b+c

## Текст программы

```
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
bool switch_op(bool a, bool b, char op)
{
    switch (op)
    {
        case '+':
            return a || b;
        case '*':
            return a && b;
    }
}
void check(bool *pid, int size, char *buf)
{
    bool result; // Результат сравнения
    char chars[3];
    char *smb1 = "+";
    char *smb2 = "*";
    if ((buf[0] == *smb1) & (buf[1] == *smb2))
    {
        result = switch_op(pid[1], pid[2], buf[1]);
        result = switch_op(result, pid[0], buf[0]);
    }
    else
    {
        result = switch_op(pid[0], pid[1], buf[0]);
        result = switch_op(result, pid[2], buf[1]);
    }
}
```

```

        // Отметка, что выражение истинно
        if (result)
            printf("*");
        printf("\n");
    }
int main()
{
    const int n = 3; // Количество логических переменных
    bool pid[n];     // Массив логических переменных
    char buf[n];     // Массив логических операций
    int idx = 0;     // Текущий индекс в buf
    for (int i = 0; i < n * 2 - 1; i++)
    {
        char c = getchar();
        if (i % 2 == 1)
        {
            buf[idx] = c;
            idx++;
        }
    }
    buf[idx] = '\0';
    // Генерация последовательности из 0 и 1 и вывод на экран
    for (int i = 0; i < n; i++)
    {
        pid[i] = !!fork();
        printf("%d ", pid[i]);
    }
    // Проверка истинности выражения
    check(pid, n, buf);
}

```