|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное автономное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

|  |  |
| --- | --- |
| ФАКУЛЬТЕТ | ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ |
| КАФЕДРА | ПРИКЛАДНАЯ ИНФОРМАТИКА |

**ОТЧЕТ**

**ПО ДОМАШНЕМУ ЗАДАНИЮ**

***НА ТЕМУ:***

|  |
| --- |
| ***Системы счисления, двоичная арифметика,*** |
| ***логические функции, Bash*** |
|  |
|  |
|  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Студент | ИУ1-12Б |  |  |  | Р.О. Волков |
|  | (группа) |  | (подпись, дата) |  | (И.О. Фамилия) |
| Руководитель курсовой работы |  |  |  |  |  |
|  | П.А. Панилов |
|  |  |  | (подпись, дата) |  | (И.О. Фамилия) |
|  |  |  |  |  |  |

*2025 г.*

**Вариант 5**

1. Условие

Переведите число 100101110110.1112 в десятичную систему, затем выполните обратное преобразование через восьмеричную и шестнадцатеричную системы, проверяя точность на каждом этапе.

**Решение**

Используем развёрнутую запись числа и решим в десятичной системе

100101110110.1112= 1\*211 + 0\*210 + 0\*29 + 1\*28 + 0\*27 + 1\*26 + 1\*25 + 1\*24 + 0\*23 + 1\*22 + 1\*21 + 0\*20 + 1\*2-1 + 1\*2-2 + 1\*2-3 = 2048 + 256 + 64 + 32 + 16 + 4 + 2 + 0.5 + 0.25 + 0.125 = 2422.87510

Выполним обратное преобразование в восьмеричную систему

Для этого разобьём двоичную запись на группы по 3 разряда и заменим их цифрой восьмеричной системы

100 101 110 110.1112 = 4566.78

Проверим точность переводом в десятичную систему

4566.78 = 4\*83 + 5\*82 + 6\*81 + 6\*80 + 7\*8-1 = 2048 + 320 + 48 + 6 + 0.875 = 2422.87510

Теперь выполним обратное преобразование в шестнадцатеричную систему

Для этого разобьём двоичную запись на группы по 4 разряда, дополняя крайние неполные группы нулями, и заменим их цифрой шестнадцатеричной системы

1001 0111 0110.11102 = 976.E16

Проверим точность переводом в десятичную систему

976.E16 = 9\*162 + 7\*161 + 6\*160 + 14\*16-1 = 2304 + 112 + 6 + 0.875 = 2422.87510

2. Условие

Переведите число 2C.4F16 в двоичную и десятичную системы, а затем выполните обратное преобразование в восьмеричную.

**Решение**

Для перевода из шестнадцатеричной системы в двоичную представим каждую цифру как группу из 4ёх разрядов в двоичной и уберём лишние нули

2C.4F16 = 0010 1100.0100 1111 = 101100.010011112

Переведём число в десятичную систему с помощью развёрнутой записи

2C.4F16 = 2\*161 + 12\*160 + 4\*16-1 + 15\*16-2 = 32 + 12 + 0.25 + 0,05859375 = 44.305859375

Выполним обратное преобразование в восьмеричную систему

Для этого разобьём двоичную запись на группы по 3 разряда, дополняя крайние неполные группы нулями, и заменим их цифрой восьмеричной системы

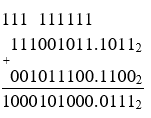
101 100.010 011 1102 = 54.2368

3. Условие

Выполните сложение чисел 111001011.10112+1011100.112 и переведите результат в шестнадцатеричную систему. Затем выполните вычитание с числом A14B16 и переведите обратно в двоичную.

**Решение**

Выполним поразрядное сложение чисел дополнив недостающие разряды нулями



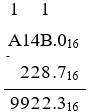
Переведём результат в шестнадцатеричную систему

Для этого разобьём двоичную запись на группы по 4 разряда, дополняя крайние неполные группы нулями, и заменим их цифрой шестнадцатеричной системы

0010 0010 1000.01112 = 228.716

Т.к. 228.716 < A14B16, их разность будет отрицательной.

Выполним поразрядное вычитание для получения модуля разности



Для перевода из шестнадцатеричной системы в двоичную представим каждую цифру как группу из 4ёх разрядов в двоичной

-9922.316 = -1001 1001 0010 0010.0011 = -1001100100100010.00112

4. Условие

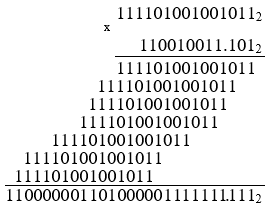
Переведите числа 7A4B16 и 623.58 в двоичную систему и выполните умножение между ними. Переведите результат в десятичную и обратно в шестнадцатеричную систему.

**Решение**

Для перевода из шестнадцатеричной и восьмеричной систем в двоичную представим каждую цифру как группу из 4ёх и 3ёх разрядов в двоичной соответственно

7A4B16 = 0111 1010 0100 10112 = 1111010010010112

623.58 = 110 010 011.1012 = 110010011.1012



Распишем развёрнутую запись для перевода в десятичную систему, перед этим добавим и вычтем единицу для более короткой записи

110000001101000001111111.1112 + 1 – 1 = 110000001101000010000000.1112 – 1 = 1\*223 + 1\*222 + 1\*215 + 1\*214 + 1\*212 + 1\*27 + 1\*2-1 + 1\*2-2 + 1\*2-3 – 1 = 8388608 + 4194304 + 32768 + 16384 + 4096 + 128 + 0.5 + 0.25 + 0.125 – 1 = 12636287.87510

Теперь переведём это в шестнадцатеричную систему, разбив двоичную запись на группы по 4 разряда и заменив каждый на цифру в шестнадцатеричной системе

1100 0000 1101 0000 0111 1111.11102 = C0D07F.E16

5. Найдите значение выражения ((4D.7B16+725.128)×(101110.11012−111001.012))÷(C2A16+1011010.0112). Результат переведите в десятичную и восьмеричную системы, затем обратно в двоичную.

Переведём каждое слагаемое в двоичную систему

4D.7B16 = 0100 1101.0111 1011 = 1001101.011110112

725.128 = 111 010 101.001 010 = 111010101.001012

C2A16 = 1100 0010 1010 = 1100001010102

Посчитаем суммы и разности в каждом множителе

1001101.011110112 + 111010101.001012 = 1000100010.101000112

101110.11012 − 111001.012 = –(111001.01 – 101110.1101) = –1010.01112

1100001010102 + 1011010.0112 = 110010000100.0112

Переведём каждый множитель в десятичную систему

1000100010.101000112 = 1\*29 + 1\*25 + 1\*21 + 1\*2-1 + 1\*2-3 + 1\*2-7 + 1\*2-8 = 512 + 32 + 2 + 0.5 + 0.125 + 0,0078125 + 0.00390625 = 546.6367187510

–1010.01112 = –(1\*23 + 1\*21 + 1\*2-2 + 1\*2-3 + 1\*2-4) = –(8 + 2 + 0.25 + 0.125 + 0.0625) = –10.437510

110010000100.0112 = 1\*211 + 1\*210 + 1\*27 + 1\*22 + 1\*2-2 + 1\*2-3= 2048 + 1024 + 128 + 4 + 0.25 + 0.125 = 3204.37510

Выполним умножение

546.6367187510 \* (–10.437510) = –5705.52075195312510

Выполним деление

–5705.52075195312510 / 3204.37510 ≈ –1.780540901721279500682660425210

Переведём целую и дробную части результата в восьмеричную систему

–110 = –18

Что бы перевести дробную часть, домножим переводимое число на основание системы, в которую происходит перевод

–0.780540901721279500682660425210 \* 8 = –6.244327213770236005461283401610

Целая часть результата является первым разрядом после запятой переведённого числа

Повторим умножение дробной части для получения следующих разрядом

–0.244327213770236005461283401610 \* 8 = –1.954617710161888043690267212810

–0,954617710161888043690267212810 \* 8 = –7,636941681295104349522137702410

–0,636941681295104349522137702410 \* 8 = –5,095533450360834796177101619210

–0,095533450360834796177101619210 \* 8 = –0,764267602886678369416812953610

–0,764267602886678369416812953610 \* 8 = –6,114140823093426955334503628810

–0,114140823093426955334503628810 \* 8 = –0,913126584747415642676029030410

–0,913126584747415642676029030410 \* 8 = –7,305012677979325141408232243210

–0,305012677979325141408232243210 \* 8 = –2,440101423834601131265857945610

Сочтём точность приемлемой и соберём разряды результата воедино

–1.780540901721279500682660425210 = –1.6175060728

Переведём это в двоичную систему

–1.6175060728 = –001.110 001 111 101 000 110 000 111 010 = –1.110001111101000110000111012

7. Условие

Используя мультиплексор с 4 входами и двумя адресными линиями A1,A0 создайте устройство, которое выдает на выход Q значение 1, если активен D0 и одновременно хотя бы один из входов D1 или D3 также активен, и 0 в противном случае.

**Решение**

Мультиплексор с 4мя входами выбирает вход D0, D1, D2 или D3 в зависимости от адресных линий A1 и A0 и отправляет его на выход Q

|  |  |  |
| --- | --- | --- |
| **A1** | **A0** | **Q** |
| 0 | 0 | D0 |
| 0 | 1 | D1 |
| 1 | 0 | D2 |
| 1 | 1 | D3 |

Устройство должно выдавать 1, если D0 активен вместе с D1 или D3, и 0 в остальных случаях.

Q = 1, только если вход D0 активен, значит он должен быть равен 1

Для того, чтобы выход был равен 1 при активных входах D1 или D3 можно использовать функцию

Q = (¬A1 x A0 x D1) + (A1 x A0 x D3)

Построим таблицу истинности функции

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A1** | **A0** | **D1** | **D3** | **Q** |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | — | — | 0 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 |

8. Условие

С помощью мультиплексора с 8 входами и тремя адресными линиями A2,A1,A0 создайте устройство, которое выдает на выход Q значение 1, если хотя бы один из входов D1, D4, или D7 активен, и 0 в противном случае, но если активны D0 и D6 одновременно, то устройство должно выдать 0 независимо от остальных входов.

**Решение**

Мультиплексор с 8мя входами выбирает вход D0, D1, D2, D3, D4, D5, D6 или D7 в зависимости от адресных линий A2, A1 и A0 и отправляет его на выход Q

|  |  |  |  |
| --- | --- | --- | --- |
| **A2** | **A1** | **A0** | **Q** |
| 0 | 0 | 0 | D0 |
| 0 | 0 | 1 | D1 |
| 0 | 1 | 0 | D2 |
| 0 | 1 | 1 | D3 |
| 1 | 0 | 0 | D4 |
| 1 | 0 | 1 | D5 |
| 1 | 1 | 0 | D6 |
| 1 | 1 | 1 | D7 |

Что бы устройство выдало 1, нужно, чтобы входы D0 и D6 не были активны одновременно, а хотя бы один из D1, D4 или D7 был активен

Для этого можно использовать функцию

Q = (¬D0 + ¬D6) \* (D1 + D4 + D7) \* ((¬A2 \* ¬A1 \* A0) + (A2 \* ¬A1 \* ¬A0) + (A2 \* A1 \* A0))

Построим таблицу истинности функции

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **A2** | **A1** | **A0** | **D0** | **D1** | **D4** | **D6** | **D7** | **Q** |
| — | — | — | 1 | — | — | 1 | — | 0 |
| 0 | 0 | 1 | 0 | 1 | — | 0 | — | 1 |
| 0 | 1 | 0 | — | — | — | — | — | 0 |
| 0 | 1 | 1 | — | — | — | — | — | 0 |
| 1 | 0 | 0 | 0 | — | 1 | 0 | — | 1 |
| 1 | 0 | 1 | — | — | — | — | — | 0 |
| 1 | 1 | 0 | — | — | — | — | — | 0 |
| 1 | 1 | 1 | 0 | — | — | 0 | 1 | 1 |

Напишите Bash-скрипт, который:

1. Принимает аргумент - имя файла. Проверяет, существует ли файл. Если нет, создает файл и записывает в него 10 случайных строк.

2. В цикле читает строки из файла и отправляет их на локальный сервер с помощью netcat.

3. После завершения работы netcat выводит сообщение о том, сколько строк было отправлено.

4. Запускает фоновый процесс, который записывает текущее время в файл log.txt каждую минуту.

Решение

if [ ! -f $1 ]; then #проверяем существование файла

touch $1 #создаём файл в случае его отсуствия

for i in {1..10}; do

echo $RANDOM >> $1 #наполняем созданный файл 10тью случайными строками

done

fi

sed -n -e p $1 | ncat localhost 733 #читаем каждую строку и выводим в netcat

cat $1 | wc -l #считаем количество строк в файле, то есть количество строк выведенных в netcat

while true;do #создаём фоновый процесс

date > log.txt #записываем время в файл

sleep 60 #ждём минуту до следующей записи

done