МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

"МИРЭА - Российский технологический университет"

РТУ МИРЭА

Институт информационных технологий (ИТ)
Кафедра математического обеспечения и стандартизации информационных технологий (МОСИТ)

ОТЧЁТ ПО ПРАКТИЧЕСКОМУ ЗАДАНИЮ №1

по дисциплине «Структуры и алгоритмы обработки данных»

Тема: «Поразрядные операции и их применение»

Отчет представлен к рассмотрению:
Студент группы ИНБО-01-20 «14» сентября 2021 г. _____ Салов В.Д.
(подпись)

Преподаватель «14» сентября 2021 г. _____ Сорокин А.В.
(подпись)

Москва, 2021 г.

СОДЕРЖАНИЕ

Цель работы

Получение навыков применения поразрядных операций в алгоритмах.

Постановка задачи

Выполнить упражнения по применению битовых операций по изменению значений битов в ячейке оперативной памяти, созданию маски для изменения значения ячейки. Создать выражение, содержащее поразрядные операции для выполнения определенной операции над значением ячейки. Выполнить тестирование программы и оформить отчёт.

Вариант 7.

Задание 1.

Разработать программу, которая продемонстрирует выполнение упражнений варианта. Результаты выполнения упражнения выводить на монитор.

Упражнение 1.

Определить переменную целого типа, присвоить ей значение, используя константу в шестнадцатеричной системе счисления. Разработать оператор присваивания и его выражение, которое установит **четыре старших бита** исходного значения переменной в значение **1**, используя соответствующую маску и поразрядную операцию.

Код программы:

```
x = int(input("Введите 16-битное число в 16 с/с: "), 16)
mask = int("1111000000000000", 2)
print()
print("Получено число:", hex(x))
print("Используется маска:", hex(mask))
print("Найден результат:", hex(x | mask))
```

Рисунок 1 – Код программы для упражнения 1.

Чтобы установить **четыре старших бита** исходного значения переменной ***x*** в значение **1**, можно использовать маску **0xF000** и применить **операцию побитового «ИЛИ»** для переменной ***x*** с данной маской. В результате в старших четырёх битах установится значение **1**, а остальные биты останутся без изменений.

Тестирование кода программы:

```
Введите 16-битное число в 16 с/с: AFFF

Получено число: 0xafff
Используется маска: 0xf000
Найден результат: 0xffff
```

Рисунок 2 – Тестирование кода программы для упражнения 1.

На вход поступает число **0xAFFF**. После выполнения **операции побитового «ИЛИ»** для переменной ***x*** с маской **0xF000** первый и третий бит переменной ***x*** меняют своё значение на **1**, отсюда конечное значение переменной ***x***: **0xFFFF**.

Упражнение 2.

Определить переменную целого типа. Разработать оператор присваивания и его выражение, которое **обнуляет 9-ый, 11-ый и 3-ий биты** исходного значения переменной, используя соответствующую маску и поразрядную операцию. Значение в переменную вводится с клавиатуры.

Код программы:

```
x = int(input("Введите 16-битное число в 16 с/с: "), 16)
mask = int("1111010111110111", 2)
print()
print("Получено число:", hex(x))
print("Используется маска:", hex(mask))
print("Найден результат:", hex(x & mask))
```

Рисунок 3 – Код программы для упражнения 2.

Чтобы обнулить **заданные биты** исходного значения переменной ***x***, можно использовать маску **0xF5F7** и применить **операцию побитового «И»** для переменной ***x*** с данной маской. В результате в **заданных битах** установится значение **0**, а остальные биты останутся без изменений.

Тестирование кода программы:

```
Введите 16-битное число в 16 с/с: BFFA

Получено число: 0xbffa
Используется маска: 0xf5f7
Найден результат: 0xb5f2
```

Рисунок 4 – Тестирование кода программы для упражнения 2.

На вход поступает число **0xBFFA**. После выполнения **операции побитового «И»** для переменной ***x*** с маской **0xF5F7 9-ый, 11-ый и 3-ий биты** переменной ***x*** меняют своё значение на **0**, отсюда конечное значение переменной ***x***: **0xB5F2**.

Упражнение 3.

Определить переменную целого типа. Разработать оператор присваивания и выражение, которое **умножает значение переменной на число 512**, используя соответствующую поразрядную операцию. Изменяемое число вводится с клавиатуры.

Код программы:

```
x = int(input("Введите 16-битное число в 16 с/с: "), 16)
print()
print("Получено число:", hex(x))
print("Найден результат:", hex(x << 9))
```

Рисунок 5 – Код программы для упражнения 3.

Чтобы **умножить заданное число на 512**, необходимо произвести **битовый сдвиг влево 9 раз**.

Тестирование кода программы:

```
Введите 16-битное число в 16 с/с: AA

Получено число: 0xaa
Найден результат: 0x15400
```

Рисунок 6 – Тестирование кода программы для упражнения 3.

На вход поступает число **0xAA**. После выполнения **побитового сдвига влево в количестве 9 раз** для переменной ***x*** конечное значение переменной ***x*** становится **в 512 раз больше** исходного: **0x15400**.

Упражнение 4.

Определить переменную целого типа. Разработать оператор присваивания и выражение, которое **делит значение переменной на число 512**, указанное в четвертом столбце варианта, используя соответствующую поразрядную операцию. Изменяемое число вводится с клавиатуры.

Код программы:

```
x = int(input("Введите 16-битное число в 16 с/с: "), 16)
print()
print("Получено число:", hex(x))
print("Найден результат:", hex(x >> 9))
```

Рисунок 7 – Код программы для упражнения 4.

Чтобы **разделить заданное число на 512**, необходимо произвести **битовый сдвиг вправо 9 раз**.

Тестирование кода программы:

```
Введите 16-битное число в 16 с/с: AAAA

Получено число: 0xaaaa
Найден результат: 0x55
```

Рисунок 8 – Тестирование кода программы для упражнения 4.

На вход поступает число **0xAAAA**. После выполнения **побитового сдвига вправо в количестве 9 раз** для переменной ***x*** конечное значение переменной ***x*** становится **в 512 раз меньше** исходного: **0x55**.

Упражнение 5.

Определить переменную целого типа. Разработать оператор присваивания и выражение, в котором используются только поразрядные операции. В выражении используется маска – переменная. Маска может быть инициализирована единицей в младшем разряде (вар. 1) или единицей в старшем разряде (вар. 2). Изменяемое число вводится с клавиатуры. Необходимо **обнулить n-ый бит в 0, используя маску пункта 1 (с единицей в младшем разряде)**.

Код программы:

```
x = int(input("Введите 16-битное число в 16 с/с: "), 16)
n = int(input("Введите номер n-го бита (от 0 до 15): "))
mask = int("0000000000000001", 2)
print()
print("Получено число:", hex(x))
print("Используется маска:", hex(mask))
print("Номер обнуляемого бита:", n)
print("Найден результат:", hex(x & ~(mask << n)))
```

Рисунок 9 – Код программы для упражнения 5.

В первую очередь производится **побитовый сдвиг влево** для заданной маски столько раз, чтобы бит маски с номером **n** принял значение **1**. После этого биты маски **инвертируются**, и далее применяется **операция побитового «И»** для переменной **x** с изменённой маской. В результате **n-ый бит** переменной ***x*** обнуляется, а остальные биты данной переменной остаются неизменными.

Тестирование кода программы:

```
Введите 16-битное число в 16 с/с: FAFF
Введите номер n-го бита (от 0 до 15): 12

Получено число: 0xfaff
Используется маска: 0x1
Номер обнуляемого бита: 12
Найден результат: 0xeaff
```

Рисунок 10 – Тестирование кода программы для упражнения 5.

На вход поступает число **0xFAFF**, обнуляемым выбран **12-ый бит**. Происходит **побитовый сдвиг влево в количестве 12 раз** для заданной маски (**0x0001**), после чего **12-ый бит** маски принимает значение **1**, а все остальные биты маски – значение **0**. Изменённое значение маски: **0x1000**.

Далее происходит **инверсия битов** маски, после которой **12-ый бит** маски принимает значение **0**, а все остальные биты маски – значение **1**. Изменённое значение маски: **0xEFFF**.

В завершение применяется **операция побитового «И»** для переменной ***x*** с изменённой маской, после чего **12-ый бит** переменной ***x*** обнуляется; конечное значение переменной ***x***: **0xEAFF**.

Вывод

В ходе работы были приобретены практические навыки по использованию поразрядных операций в алгоритмах посредством выполнения упражнений по применению битовых операций.

Список информационных источников

1. Лекции по дисциплине «Структуры и алгоритмы обработки данных» / Л. А. Скворцова, МИРЭА – Российский технологический университет, 2021.
2. BestProg – [Электронный ресурс] URL: https://www.bestprog.net/ru/2019/10/21/python-bitwise-operators-ru/#q07 (Последнее обращение 11.09.2021)