**Отчет по лабораторной работе № 6** по курсу   
  
"Фундаментальная информатика"

Студент группы М8О-109Б-20 Чувилин Виталий Антонович, № по списку 19

Контакты e-mail: lemonvitaliy@mail.ru

Работа выполнена: « 10 » ноября 2020 г.   
  
Преподаватель: каф. 806 Титов В.К.

Входной контроль знаний с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Отчет сдан « » 2020 г., итоговая оценка \_\_\_\_\_

Подпись преподавателя \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **Тема:** Конструирование диаграмм Тьюринга
2. **Цель работы:** Научиться работать с диаграммами Тьюринга на JDT и представлять простейшие алгоритмы в виде диаграмм
3. **Задание** (*вариант № 34*)**:** Восстановление целого числа в восьмеричной системе счисления по  
   дополнительному коду.
4. **Оборудование**:

*Оборудование ПЭВМ студента, если использовалось:*

Процессор AMD Ryzen 5 3500U с ОП 8 Гб, НМД 256 Гб.   
Монитор встроенный  
Другие устройства не использовались

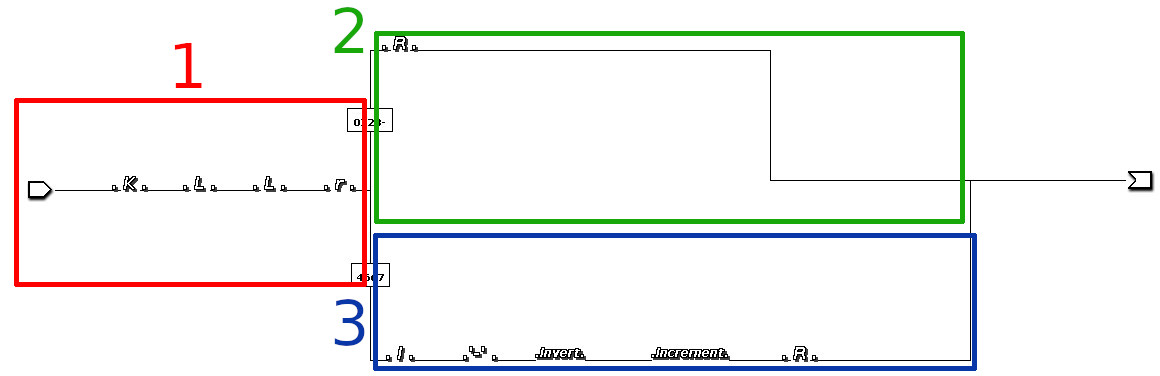
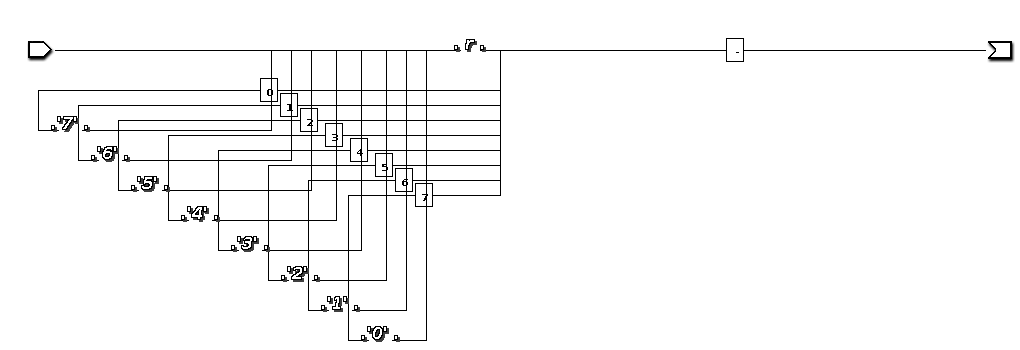
1. **Программное обеспечение:**

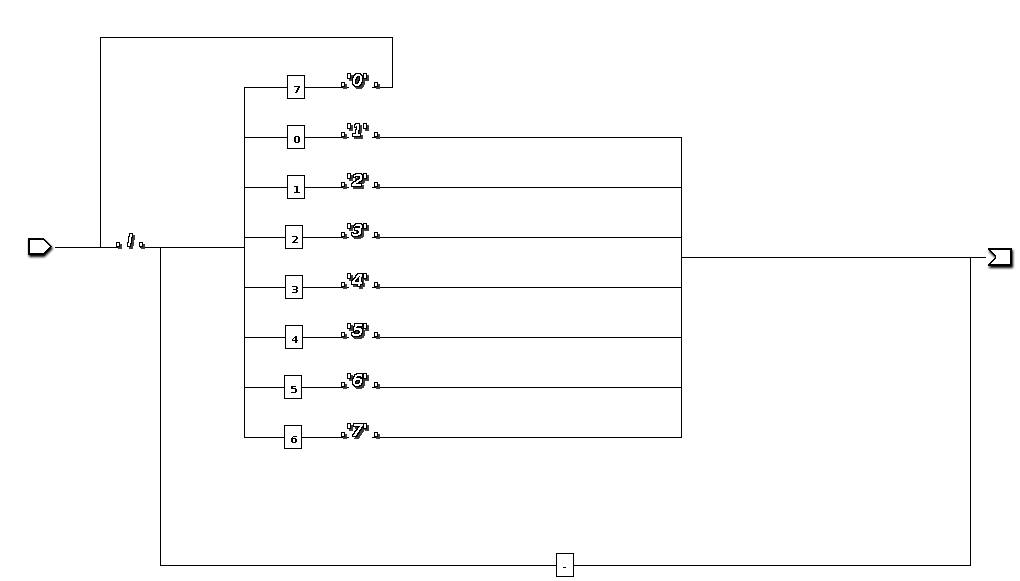
*Программное обеспечение ЭВМ студента, если использовалось:*

Операционная система семейства GNU\Linux, наименование Ubuntu 20.04 интерпретатор команд bash версия 5.0.17(1)-release  
Система программирования Не использовалась   
Редактор текстов LibreOffice

Утилиты

Местонахождение и имена файлов программ и данных

1. **Идея, метод, алгоритм**   
     
   Программа состоит из 3 частей:  
     
   1. Main   
   2. Invert   
   3. Increment   
     
     
   **I. Main –** это основная часть программы, здесь реализуется само решение.  
     
   В этой части происходит следующее:  
     
   1) Исходное слово копируется, затем курсор переносится на начало первого слова.  
   2) Если преобразования не нужны, тогда курсор возвращается в исходную позицию и программа завершает свою работу.  
   3) Если нужны преобразования, тогда устанавливаем перед словом «-» и реализуем функции «Invert» и «Increment», после чего возвращаем курсор на исходную позицию и завершаем работу программы.  
     
     
   **II. Invert –** это дополнительная часть программы, здесь реализовано инвертирование слова.  
     
     
   Так как для нахождения искомого числа исходное слово нужно инвертировать, а работаем мы в 8-ой системе счисления, то 0 заменяется на 7, 1 на 6, 2 на 5, и т. д.  
     
     
   **III. Increment –** это еще одна дополнительная часть программы, в которой реализовано увеличение результата на 1.

Здесь мы просто увеличиваем последний разряд на единицу, и, если нужно, переносим единицу в следующий разряд. **7. Сценарий выполнения работы**

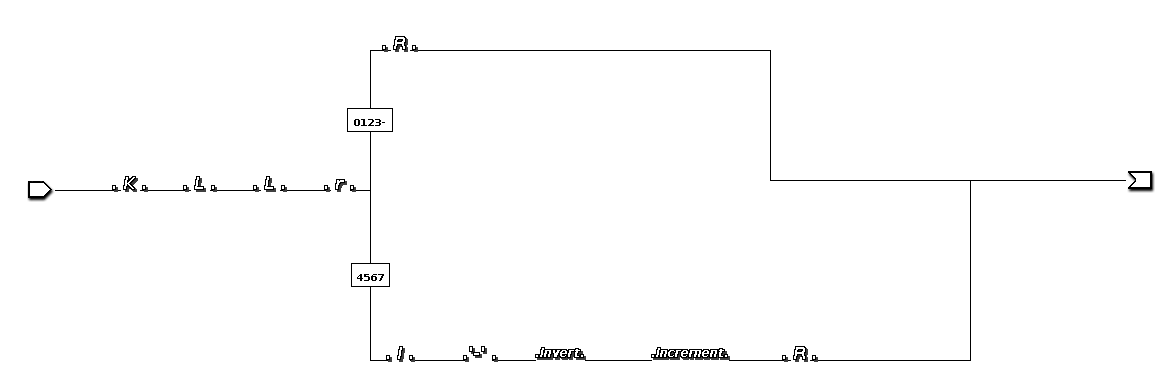
|  |  |  |
| --- | --- | --- |
| **№** | **Входные данные** | **Выходные данные** |
| 1 | 1234567 | 1234567 1234567 |
| 2 | 76543210 | -01234570 76543210 |
| 3 | 500000 | -300000 500000 |
| 4 | 37646 | 37646 37646 |
| 5 | 67534 | -10244 67534 |

**Первое число является искомым, второе исходным.**

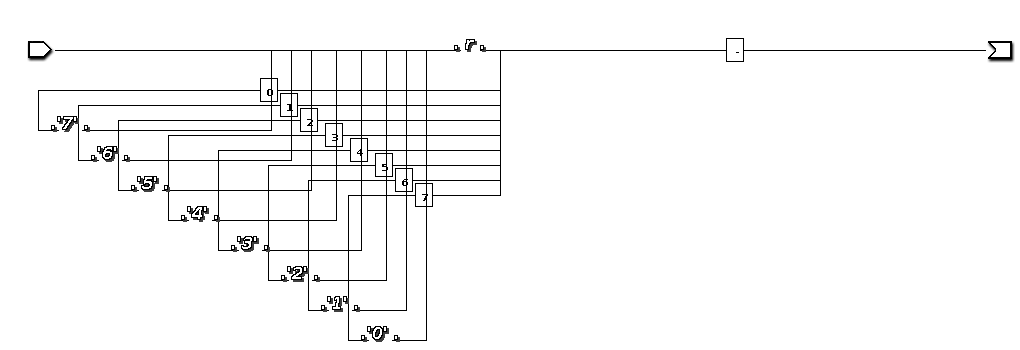
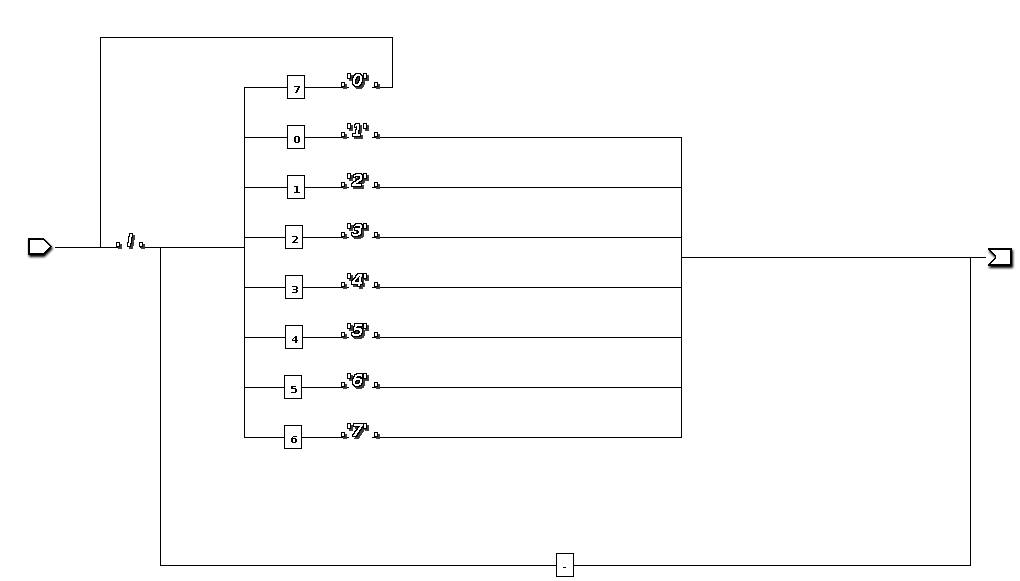
**Все числа записаны в 8-ой системе счисления.  
Случай, когда первый символ является 0 не обработан, подразумевается, что это не имеет смысла.**

*Пункты 1-7 отчета составляются сторого до начала лабораторной работы.*

*Допущен к выполнению работы.*  **Подпись преподавателя \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**8. Распечатка протокола**  (подклеить листинг окончательного варианта программы с тестовыми примерами, подписанный преподавателем)  
**Main:**

**Invert:**

**Increment:  
  
9. Д****невник отладки** должен содержать дату и время сеансов отладки и основные события (ошибки в сценарии и программе, нестандартные ситуации) и краткие комментарии к ним. В дневнике отладки приводятся сведения об использовании других ЭВМ, существенном участии преподавателя и других лиц в написании и отладке программы.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Лаб. или дом. | Дата | Время | Событие | Действие по исправлению | Примечание |
|  |  |  |  |  |  |  |

**10. Замечания автора** по существу работы: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**11. Выводы**

C диаграммами работать намного удобнее, чем с командами. В диаграммах наглядно видно какая часть программы за что отвечает, в то время как в командах понять подобное гораздо сложнее.

Недочёты при выполнении задания могут быть устранены следующим образом: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись студента \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_