Индивидуальное домашнее задание (Гринченко Евгений, БПИ 236, вариант 33)

Небольшое предисловие перед началом - код программы я реализовывал сразу на оценку 8-9, поэтому не удалось соблюдать условия итеративности. Постараюсь отразить в этом отчёте выполняемость критериев на оценку 4-7.
Финальный вариант программы, которую я реализовал, чтобы претендовать на оценку 10, был разбит на 4 ассемблерных файла:
1)Файл **data.s** в этой программе отвечает за объявление и инициализацию данных, используемых в процессе выполнения.

```
.data
start_game: .asciz "Вы хотити ввести данные с клавиатуры или запустить автоматическое тестирование(0-с клавиатуры, 1 - автоматическое тестирование) "
start_txt: .asciz "Enter the number of elements (1-10): " # Ввод числа элементов массива
result_el: .asciz "Array B element: "  # Вывод элемента массива
error_start: .asciz "Wrong number. Repeat input of varieble, please!"
end_choice: .asciz "Вы хотите закончить программу или заново пройти(0-закончить, 1 - заново пройти) "
repeat_choice: .asciz "Хочешь использовать массив B в качестве входных данных или ввести данные с клавиатуры(0 - с клавиатуры, 1 использовать B)"
txt: .asciz "Enter a number: "  # Ввод числа
ln: .asciz "\n" # Перенос строки
test: .asciz "Новый тест"
A1:  .word 1, 2, 3,4,5,6,7,8,9,10   # Массив A1
A2:  .word 1   # Массив A2
A3:  .word 1, 2, 3   # Массив A3
A4:  .word 1, 2, 3, 4, 5  # Массив A4
A5:  .word 4, 7, 10, 0, 13 #Массив A5
A6: .word 23,0,0,4 #Массив A6

.align 2

array: .space 40  # Массив A
array_res: .space 40  # Массив B (результат)
```

2)Файл `macros.inc` выполняет вспомогательные функции, которые упрощают код основной программы за счет использования макросов для повторяющихся операций.

```
.include "data.s"
.macro PRINT_STR, (%str)
```

```
    la a0, %str     # Загрузить адрес строки
    li a7, 4        # Системный вызов для вывода строки
    ecall
.end_macro

.macro INPUT_NUM
    li a7, 5        # Системный вызов для ввода числа
    ecall
.end_macro

.macro LOAD_ARRAY, (%arr), (%size)
    la t0, %arr      # Загрузить адрес массива
    addi sp, sp, -4  # Выделить место в стеке
    sw t0, 0(sp)     # Сохранить адрес массива в стеке
    li t0, %size     # Размер массива
    addi sp, sp, -4  # Выделить место в стеке
    sw t0, 0(sp)     # Сохранить размер массива в стеке
.end_macro
```

3)Файл `text.s` с подключением подпрограмм (subroutines.s) содержит основной код программы, который отвечает за управление вводом данных, выполнение автоматических тестов и обработку массивов.

```
.include "subroutines.s"  # Подключаем подпрограммы



.text
.globl main
.globl start_program

main:
  PRINT_STR start_game
  INPUT_NUM
  beqz a0, start_program
  li t1, 1
  beq a0, t1, auto_test
  PRINT_STR error_start
  PRINT_STR ln
  j main

auto_test:
    LOAD_ARRAY A1, 10
    LOAD_ARRAY A2, 1
```

```
    LOAD_ARRAY A3, 3
    LOAD_ARRAY A4, 5
    LOAD_ARRAY A5, 5
    LOAD_ARRAY A6, 4

    li t0, 12
    addi sp, sp,-4
    sw t0, 0(sp)

    li t0, 1
    addi sp, sp,-4
    sw t0, 0(sp)

    j auto_test_work


start_program:
  PRINT_STR start_txt
  INPUT_NUM
  li a1, 1
  jal check_main_one
  bnez a2,start_program
  li a1, 10
  jal check_main_ten
  bnez a2,start_program
  mv t0, a0
  la t1, array
  li t2, 0
  j work
```

4)Файл `subroutines.s` содержит основную логику программы, которая обрабатывает ввод данных, выполнение автоматических тестов и манипуляции с массивами. Он также включает управление завершением программы и повторной работой.

```
.include "macros.inc"

.text
auto_test_work:
  PRINT_STR test
  PRINT_STR ln
  lw t0, 4(sp)     # Загружаем значение счётчика (первый раз)
  slli t1, t0, 2   # Умножаем значение счётчика на 4
  add t2, sp, t1   # Рассчитываем новое смещение в стеке
```

```
  lw t4, 4(t2)     # Загружаем элемент по новому адресу
  addi t0, t0, -1
  sw t0, 4(sp)     # Обновляем значение счётчика (4(sp) вместо (sp))
  lw t0, 4(sp)     # Загружаем значение счётчика (второй раз) тут размер МАССИВА
  slli t1, t0, 2   # Умножаем значение счётчика на 4
  add t2, sp, t1   # Рассчитываем новое смещение в стеке
  lw t3, 4(t2)     # Загружаем элемент по новому адресу
  addi t0, t0, -1
  sw t0, 4(sp)     # Обновляем значение счётчика (4(sp) вместо (sp))
  li t2,0
  add t2,t3,zero
  #t2 - размер массива, t4 - указатель на начало
  add a1,t0,zero
  jal update_flag
  la t1, array       # Загрузить адрес начала массива A
  # Загрузить адрес начала массива A_номер - t4
  jal clear_A_numb
  la t1, array       # Загрузить адрес начала массива A
  li a1,0
  jal copy_to_A
  li t0,0
  add t0,a1,zero
  j end_work

update_flag:
  beqz a1, flag
  ret
flag:
  lw t1, 0(sp)
  li t1,0
  sw t1, 0(sp)
  ret

clear_A_numb:
  li t5, 0
  sw t5, 0(t1)
  addi t1, t1, 4
  addi t3, t3, -1
  bnez t3, clear_A_numb
  ret

copy_to_A:
  lw t3, 0(t4)       # Загрузить элемент из массива A1 в t1
  sw t3, 0(t1)       # Записать элемент t1 в массив A
  addi t1, t1, 4     # Увеличить указатель массива A
  addi t4, t4, 4     # Увеличить указатель массива A1
```

```
  addi t2, t2, -1    # Уменьшить счетчик элементов
  addi a1,a1,1
  bnez  t2, copy_to_A # Повторять, пока не скопируем все элементы
  ret

check_main_one:
 blt a0, a1, done
 li a2, 0
 ret
check_main_ten:
 bgt a0, a1, done
 li a2, 0
 ret
done:
  li a2, 1
  PRINT_STR error_start
  PRINT_STR ln
  ret
work:
  beq t0, t2, end_work    # Если все элементы введены, завершить цикл
  PRINT_STR txt           # Вывод приглашения для ввода числа
  INPUT_NUM               # Ввод числа в массив
  sw a0, (t1)             # Сохранение числа в массив
  addi t2, t2, 1          # Увеличение счётчика
  addi t1, t1, 4          # Переход к следующему элементу массива
  j work

end_work:
  li t2, 0                # Инициализация счётчика для обработки массива B
  la t1, array            # Указатель на начало массива A
  la t3, array_res        # Указатель на начало массива B
  li t4, 0                # Предыдущий элемент массива A
  li t5, 0                # Текущий элемент массива A
  li t6, 1                # Количество элементов для обработки
  beq t0, t6, one_element # Если только один элемент, переходим к обработке одного элемента
  j tmp_program

tmp_program:
  li t6, 0
  lw t4, (t1)             # Загрузка первого элемента массива A
  addi t2, t2, 1          # Увеличение счётчика элементов
  addi t1, t1, 4          # Переход к следующему элементу массива A
program:
  beq t0, t2, end_subst   # Если обработаны все элементы, завершить цикл
  lw t5, (t1)             # Загрузка следующего элемента массива A
  sub t6, t5, t4          # Разность текущего элемента и предыдущего
```

```
  sw t6, (t3)            # Сохранение разности в массив B
  mv t4, t5              # Обновление предыдущего элемента для следующего шага
  addi t3, t3, 4         # Переход к следующему элементу массива B
  addi t2, t2, 1         # Увеличение счётчика
  addi t1, t1, 4         # Переход к следующему элементу массива A
  j program

one_element:
  lw t5, (t1)            # Обработка одного элемента
  addi t2, t2, 1
  addi t1, t1, 4
  sw t5, (t3)
  j print_one

print_one:
  lw t5, (t3)            # Вывод одного элемента массива B
  addi t3, t3, 4
  PRINT_STR result_el
  mv a0, t5
  li a7, 1
  ecall
  PRINT_STR ln
  lw t0, 0(sp)     # Загружаем значение счётчика (первый раз)
  bnez t0,auto_test_work
  li t2, 1
  j game_over

end_subst:
  addi t0, t0, -1        # Уменьшение количества элементов
  li t2, 0
  li t5, 0
  la t3, array_res       # Указатель на начало массива B
  j start_print

start_print:
  beq t0, t2, completion   # Если все элементы выведены, завершить
  lw t5, (t3)            # Загрузка элемента из массива B
  mv a1, t5              # Передаем элемент через регистр a1
  jal print              # Вызов подпрограммы print с передачей параметра через регистр
  addi t2, t2, 1
  addi t3, t3, 4
  j start_print

print:
  PRINT_STR result_el
  # В подпрограмму передаем элемент для вывода через регистр a1
```

```
  mv a0, a1              # Вывод числа, переданного через регистр a1
  li a7, 1
  ecall
  PRINT_STR ln
  ret                    # Возврат из подпрограммы

completion:
 lw t1, 0(sp)     # Загружаем значение счётчика (первый раз)
 beqz t1,game_over
 la t3, array_res
 li t4, 10
 jal clear_B
 j auto_test_work

game_over:
 PRINT_STR end_choice
 INPUT_NUM
 beqz a0,end
 li t1,1
 beq a0,t1,repeat
 PRINT_STR error_start
 PRINT_STR ln
 j game_over
repeat:
 PRINT_STR ln
 PRINT_STR repeat_choice
 INPUT_NUM
 beqz a0, main
 li t1,1
 beq a0,t1,transformation
 PRINT_STR error_start
 PRINT_STR ln
 j repeat



transformation:
  li a1,0
  jal main_repeat            # Переход к функции main
  li t0,0
  add t0,a1,zero
  j end_work

main_repeat:
    # Указатели на массивы
    la t1, array       # Начало массива A в t1
    la t3, array_res   # Начало массива B в t3
```

```
    li t4, 10          # Максимальное количество элементов (10)

    # Очистить массив A
clear_A:
    li t5, 0
    sw t5, 0(t1)
    addi t1, t1, 4
    addi t4, t4, -1
    bnez t4, clear_A

    # Восстановить указатели
    la t1, array
    la t3, array_res
    li t4, 0
    add t4,t2,zero

    # Копировать массив B в массив A
load_B_to_A:
    lw t5, 0(t3)
    sw t5, 0(t1)
    addi t1, t1, 4
    addi t3, t3, 4
    addi t4, t4, -1
    addi a1,a1,1
    bnez t4, load_B_to_A

    # Очистить массив B
    la t3, array_res
    li t4, 10
clear_B:
    li t5, 0
    sw t5, 0(t3)
    addi t3, t3, 4
    addi t4, t4, -1
    bnez t4, clear_B
    # Возврат в метку transformation
    ret
end:
  li a7, 10              # Завершение программы
  ecall
```

Моя программа должна формировать массив B из элементов массива A, путем записи разности между двумя соседним элементами. В начале программы у пользователя запрашивается количество элементов, которые будут в массиве A(от 1 до 10) - есть проверка, что пользователь

ввёл корректные данные(число находится в диапазоне от 1 до 10), иначе повтор ввода. После ввода всех элементов массива A, массив B заполняется разностями между текущим и предыдущим элементами массива A. Я рассмотрел крайний случай, когда в массиве A один элемент. В таком случае, я просто записываю этот элемент в массив B.

t1 - указатель на начало массива A

t2 - инициализация счётчика для обработки массива B

t3 - указатель на начало массива B

t4 - предыдущий элемент массива A

t5 - текущий элемент массива A

t6 - количество элементов для обработки

```
program:
 beq t0, t2, end_subst   # Если обработаны все элементы, завершить цикл
 lw t5, (t1)             # Загрузка следующего элемента массива A
 sub t6, t5, t4          # Разность текущего элемента и предыдущего
 sw t6, (t3)             # Сохранение разности в массив B
 mv t4, t5               # Обновление предыдущего элемента для следующего шага
 addi t3, t3, 4          # Переход к следующему элементу массива B
 addi t2, t2, 1          # Увеличение счётчика
 addi t1, t1, 4          # Переход к следующему элементу массива A
 j program
```

```
Enter the number of elements (1-10): -1
Wrong number. Repeat input of varieble, please!
Enter the number of elements (1-10): 11
Wrong number. Repeat input of varieble, please!
Enter the number of elements (1-10): 4
Enter a number: 2
Enter a number: 4
Enter a number: 6
Enter a number: 8
Array B element: 2
Array B element: 2
Array B element: 2
```

Крайний случай

```
one_element:
  lw t5, (t1)           # Обработка одного элемента
  addi t2, t2, 1
  addi t1, t1, 4
  sw t5, (t3)
  j print_one
```

```
Enter the number of elements (1-10): 1
Enter a number: 34
Array B element: 34
```

Ввод данных осуществляется с клавиатуры с помощью макроса `INPUT_NUM`, а вывод данных на экран — через макрос `PRINT_STR`.

```
.include "data.s"
.macro PRINT_STR, (%str)
    la a0, %str     # Загрузить адрес строки
    li a7, 4        # Системный вызов для вывода строки
    ecall
.end_macro

.macro INPUT_NUM
    li a7, 5        # Системный вызов для ввода числа
    ecall
.end_macro
```

Было реализовано автоматическое тестирование.
В начале программы выводится строка с предложением пользователю выбрать режим работы: ввод данных с клавиатуры или автоматическое тестирование.

```
Вы хотити ввести данные с клавиатуры или запустить автоматическое тестирование(0-с клавиатуры, 1 - автоматическое тестирование)
```

```
main:
  PRINT_STR start_game
  INPUT_NUM
  beqz a0, start_program
  li t1, 1
  beq a0, t1, auto_test
  PRINT_STR error_start
  PRINT_STR ln
  j main
```

Реализована проверка корректности вводимого значения

```
Вы хотити ввести данные с клавиатуры или запустить автоматическое тестирование(0-с клавиатуры, 1 - автоматическое тестирование) 2
Wrong number. Repeat input of varieble, please!
Вы хотити ввести данные с клавиатуры или запустить автоматическое тестирование(0-с клавиатуры, 1 - автоматическое тестирование) -1
Wrong number. Repeat input of varieble, please!
Вы хотити ввести данные с клавиатуры или запустить автоматическое тестирование(0-с клавиатуры, 1 - автоматическое тестирование)
```

- Когда пользователь выбирает автоматическое тестирование, программа загружает несколько предопределенных массивов (A1, A2, A3, A4, A5, A6), которые содержат различные наборы

чисел и различаются по размеру.

```
A1:   .word 1, 2, 3,4,5,6,7,8,9,10    # Массив A1
A2:   .word 1    # Массив A2
A3:   .word 1, 2, 3    # Массив A3
A4:   .word 1, 2, 3, 4, 5   # Массив A4
A5:   .word 4, 7, 10, 0, 13 #Массив A5
A6: .word 23,0,0,4 #Массив A6
```

- Каждый массив загружается с помощью макроса `LOAD_ARRAY`, который сохраняет в стек адрес массива и его размер, также счётчик элементов - в данном случае 12 и флаг(1 - тестовые данные ещё есть, 0 - все тестовые данные обработаны.

```
.macro LOAD_ARRAY, (%arr), (%size)
    la t0, %arr      # Загрузить адрес массива
    addi sp, sp, -4  # Выделить место в стеке
    sw t0, 0(sp)     # Сохранить адрес массива в стеке
    li t0, %size     # Размер массива
    addi sp, sp, -4  # Выделить место в стеке
    sw t0, 0(sp)     # Сохранить размер массива в стеке
.end_macro
```

```
auto_test:
    LOAD_ARRAY A1, 10
    LOAD_ARRAY A2, 1
    LOAD_ARRAY A3, 3
    LOAD_ARRAY A4, 5
    LOAD_ARRAY A5, 5
    LOAD_ARRAY A6, 4

    li t0, 12
    addi sp, sp,-4
    sw t0, 0(sp)

    li t0, 1
    addi sp, sp,-4
    sw t0, 0(sp)
    j auto_test_work
```

После загрузки массивов программа переходит в цикл `auto_test_work`, где происходит обработка каждого массива. В начале с помощью вызова макросов выводятся строки на экран - сообщение о начале тестирования и перевод на новую строку

Далее считываем из стека количество элементов массива и указатель на начало нашего массива
Значение счётчика уменьшается на 2(один раз после считывания размера массива и ещё один раз после считывания самого массива), и новое значение записывается обратно в стек.
Есть подпрограмма `update_flag`. Эта подпрограмма проверяет, равно ли значение `a1` нулю. Если равно, она устанавливает новое значение для флага (обнуляет значение в стеке).
Используются подпрограммы, такие как `clear_A_numb`, `copy_to_A`, которые не используют параметры. Они решают задачи очистки массивов и копирования данных.
Подпрограмма `clear_A_numb` очищает массив A, устанавливая все его элементы в 0. Она использует регистр `t1` для указания на текущий элемент и `t3` в качестве счётчика.
Программа инициализирует массив A, копируя данные из текущего тестового массива в массив A, используя процедуру `copy_to_A`.
Когда массив загружен, программа начинает вычислять разности между соседними элементами массива A, сохраняя результат в массив B.

```
auto_test_work:
  PRINT_STR test
  PRINT_STR ln
  lw t0, 4(sp)     # Загружаем значение счётчика (первый раз)
  slli t1, t0, 2   # Умножаем значение счётчика на 4
  add t2, sp, t1   # Рассчитываем новое смещение в стеке
  lw t4, 4(t2)     # Загружаем элемент по новому адресу
  addi t0, t0, -1
  sw t0, 4(sp)     # Обновляем значение счётчика (4(sp) вместо (sp))
  lw t0, 4(sp)     # Загружаем значение счётчика (второй раз) тут размер МАССИВА
  slli t1, t0, 2   # Умножаем значение счётчика на 4
  add t2, sp, t1   # Рассчитываем новое смещение в стеке
  lw t3, 4(t2)     # Загружаем элемент по новому адресу
  addi t0, t0, -1
  sw t0, 4(sp)     # Обновляем значение счётчика (4(sp) вместо (sp))
  li t2,0
  add t2,t3,zero
  #t2 - размер массива, t4 - указатель на начало
  add a1,t0,zero
  jal update_flag
  la t1, array       # Загрузить адрес начала массива A
  # Загрузить адрес начала массива A_номер - t4
  jal clear_A_numb
  la t1, array       # Загрузить адрес начала массива A
  li a1,0
  jal copy_to_A
  li t0,0
  add t0,a1,zero
  j end_work
```

```
update_flag:
  beqz a1, flag
  ret
flag:
  lw t1, 0(sp)
  li t1,0
  sw t1, 0(sp)
  ret
```

```
Вы хотити ввести данные с клавиатуры или запустить автоматическое тестирование(0-с клавиатуры, 1 - автоматическое тестирование) 1
Новый тест
Array B element: 1
Array B element: 1
Array B element: 1
Array B element: 1
Array B element: 1
Array B element: 1
Array B element: 1
Array B element: 1
Array B element: 1
Новый тест
Array B element: 1
Новый тест
Array B element: 1
Array B element: 1
Новый тест
Array B element: 1
Array B element: 1
Array B element: 1
Array B element: 1
Новый тест
Array B element: 3
Array B element: 3
Array B element: -10
Array B element: 13
Новый тест
Array B element: -23
Array B element: 0
Array B element: 4
```

По завершению обработки текущего массива программа либо переходит к следующему тестовому массиву, либо завершает тестирование.

```
completion:
lw t1, 0(sp)     # Загружаем значение флага (первый раз)
beqz t1,game_over
la t3, array_res
li t4, 10
jal clear_B
j auto_test_work
```

Если значение флага равно 0, то значит мы обработали все тесты и можем переходить к завершению программы.

Также был реализован повтор решения и возможность использования итогового массива B в качестве входного массива A и повторный запуск решения.
После завершения итерации программы, у пользователя будет выбор - завершить работу или начать заново.(0 - закончить, 1 - запустить новую итерацию программы). Также была реализован проверка корректности входных данных.

```
Вы хотите закончить программу или заново пройти(0-закончить, 1 - заново пройти) 3
Wrong number. Repeat input of varieble, please!
Вы хотите закончить программу или заново пройти(0-закончить, 1 - заново пройти) 4
Wrong number. Repeat input of varieble, please!
Вы хотите закончить программу или заново пройти(0-закончить, 1 - заново пройти) -2
Wrong number. Repeat input of varieble, please!
Вы хотите закончить программу или заново пройти(0-закончить, 1 - заново пройти) 0

-- program is finished running (0) --
```

```
Вы хотите закончить программу или заново пройти(0-закончить, 1 - заново пройти) 0

-- program is finished running (0) --
```

Если пользователь вводит 1, то у него появляется две опции - использовать массив B в качестве массива A или просто ввести новые данные с клавиатуры).

```
Вы хотите закончить программу или заново пройти(0-закончить, 1 - заново пройти) 1

Хочешь использовать массив B в качестве входных данных или ввести данные с клавиатуры(0 - с клавиатуры, 1 использовать B)
```

Также предусмотрена проверка корректности ввода.

```
Хочешь использовать массив B в качестве входных данных или ввести данные с клавиатуры(0 - с клавиатуры, 1 использовать B)0
Вы хотити ввести данные с клавиатуры или запустить автоматическое тестирование(0-с клавиатуры, 1 - автоматическое тестирование) 4
Wrong number. Repeat input of varieble, please!
Вы хотити ввести данные с клавиатуры или запустить автоматическое тестирование(0-с клавиатуры, 1 - автоматическое тестирование) 2
Wrong number. Repeat input of varieble, please!
Вы хотити ввести данные с клавиатуры или запустить автоматическое тестирование(0-с клавиатуры, 1 - автоматическое тестирование) -3
Wrong number. Repeat input of varieble, please!
Вы хотити ввести данные с клавиатуры или запустить автоматическое тестирование(0-с клавиатуры, 1 - автоматическое тестирование) -2
Wrong number. Repeat input of varieble, please!
```

Если пользователь вводит 0, то он снова перейдёт в начало программы, где его снова спросят о формате тестирования - либо ввод с клавиатуры либо запуск автоматических тестов.

```
Хочешь использовать массив B в качестве входных данных или ввести данные с клавиатуры(0 - с клавиатуры, 1 использовать B)0
Вы хотити ввести данные с клавиатуры или запустить автоматическое тестирование(0-с клавиатуры, 1 - автоматическое тестирование)
```

Если пользователь вводит 1, то у него происходят вычисления нового массива B.

```
Array B element: -23
Array B element: 0
Array B element: 4
Вы хотите закончить программу или заново пройти(0-закончить, 1 - заново пройти) 1

Хочешь использовать массив B в качестве входных данных или ввести данные с клавиатуры(0 - с клавиатуры, 1 использовать B)1
Array B element: 23
Array B element: 4
Вы хотите закончить программу или заново пройти(0-закончить, 1 - заново пройти)
```

```
transformation:
  li a1,0
  jal main_repeat          # Переход к функции main_repeat
  li t0,0
  add t0,a1,zero
  j end_work

main_repeat:
    # Указатели на массивы
    la t1, array       # Начало массива A в t1
```

```
    la t3, array_res   # Начало массива B в t3
    li t4, 10          # Максимальное количество элементов (10)

    # Очистить массив A
clear_A:
    li t5, 0
    sw t5, 0(t1)
    addi t1, t1, 4
    addi t4, t4, -1
    bnez t4, clear_A

    # Восстановить указатели
    la t1, array
    la t3, array_res
    li t4, 0
    add t4,t2,zero

    # Копировать массив B в массив A
load_B_to_A:
    lw t5, 0(t3)
    sw t5, 0(t1)
    addi t1, t1, 4
    addi t3, t3, 4
    addi t4, t4, -1
    addi a1,a1,1
    bnez t4, load_B_to_A

    # Очистить массив B
    la t3, array_res
    li t4, 10
clear_B:
    li t5, 0
    sw t5, 0(t3)
    addi t3, t3, 4
    addi t4, t4, -1
    bnez t4, clear_B
    # Возврат в метку transformation
    ret
```

Мы в подпрограмме transformation вызываем подпрограмму main_repeat, где устанавливаем указатели на начало массивов A и B соответственно и задаем длину для выполнения.
Для начала очищаем массив A, потом копируем элементы из B в A и параллельно увеличиваем a1, после возвращения в transformation, мы загрузим возвращаемое значение a1 в регистр t0, который отвечает за размерность массива A.

Далее мы снова восстанавливаем указатель на начало массива B и переходим в `clear_B` и очищаем массив B. Далее возвращаемся в `transformation` и далее переходим к подпрограмме `end_work`, где происходит инициализация перед началом основного действия над массивом A(вычитание соседних элементов и запись новых данных в массив B).

```
end_work:
  li t2, 0              # Инициализация счётчика для обработки массива B
  la t1, array            # Указатель на начало массива A
  la t3, array_res        # Указатель на начало массива B
  li t4, 0              # Предыдущий элемент массива A
  li t5, 0              # Текущий элемент массива A
  li t6, 1              # Количество элементов для обработки
  beq t0, t6, one_element # Если только один элемент, переходим к обработке одного элемента
  j tmp_program
```

Если запустить автоматическое тестирование, а затем выбрать опцию формирования массива A на основе массива B, то в качестве массива B будет взят последний тестовый массив.

```
Вы хотити ввести данные с клавиатуры или запустить автоматическое тестирование(0-с клавиатуры, 1 - автоматическое тестирование) 1
Новый тест
Array B element: 1
Array B element: 1
Array B element: 1
Array B element: 1
Array B element: 1
Array B element: 1
Array B element: 1
Array B element: 1
Array B element: 1
Новый тест
Array B element: 1
Новый тест
Array B element: 1
Array B element: 1
Новый тест
Array B element: 1
Array B element: 1
Array B element: 1
Array B element: 1
Новый тест
Array B element: 3
Array B element: 3
Array B element: -10
Array B element: 13
Новый тест
Array B element: -23
Array B element: 0
Array B element: 4
Вы хотите закончить программу или заново пройти(0-закончить, 1 - заново пройти) 1

Хочешь использовать массив B в качестве входных данных или ввести данные с клавиатуры(0 - с клавиатуры, 1 использовать B)1
Array B element: 23
Array B element: 4
Вы хотите закончить программу или заново пройти(0-закончить, 1 - заново пройти) 0

-- program is finished running (0) --
```

Заключение

Программа полностью реализует поставленную задачу. Все требования к обработке массива A и формированию массива B были выполнены. Каждый элемент массива A корректно

обрабатывается, а результаты выводятся пользователю. Программа также успешно обрабатывает крайние случаи, демонстрируя её стабильность и надежность. Был реализован повтор решения и опция формирования нового массива A на основе массива B, также было реализовано автоматическое тестирование.

Всем добра!