Шабалкин Виталий Витальевич

БПИ-214

Вариант №33

Задача: сформировать массив B на основе элементов массива A, полученных как разность соседних элементов.

Код на C:

```
#include <stdio.h>


int main()

{

int ek = 1;

int n;

while(ek == 1){

printf("Enter n(2-100): ");

scanf("%d", &n);

if(n >= 2 && n <= 100){

ek = 0;

break;

} else {

printf("Incorrect input, try again\n");

}

}

int a [100];

int b [100];

printf("Enter array A\n");

for(int i = 0; i < n; i++){

printf("a[%d]: ", i);

scanf("%d", &a[i]);

}

printf("Generated array B:\n");
for(int j = 0; j < n - 1; j++){
b[j] = a[j] - a[j + 1];
printf("b[%d]: %d\n", j, b[j]);
}
return 0;
}
```

Код GAS:

```
        .file       "idz_na_c_1.s"
.text
.section.rodata
.LC0:
.string   "Enter n(2-100): "
.LC1:
.string   "%d"
.LC2:
.string   "Incorrect input, try again"
.LC3:
.string   "Enter array A"
.LC4:
.string   "a[%d]: "
.LC5:
.string   "Generated array B:"
.LC6:
.string   "b[%d]: %d\n"
.text
.globl    main
.type     main, @function
main:
endbr64
pushq   %rbp
movq    %rsp, %rbp
```

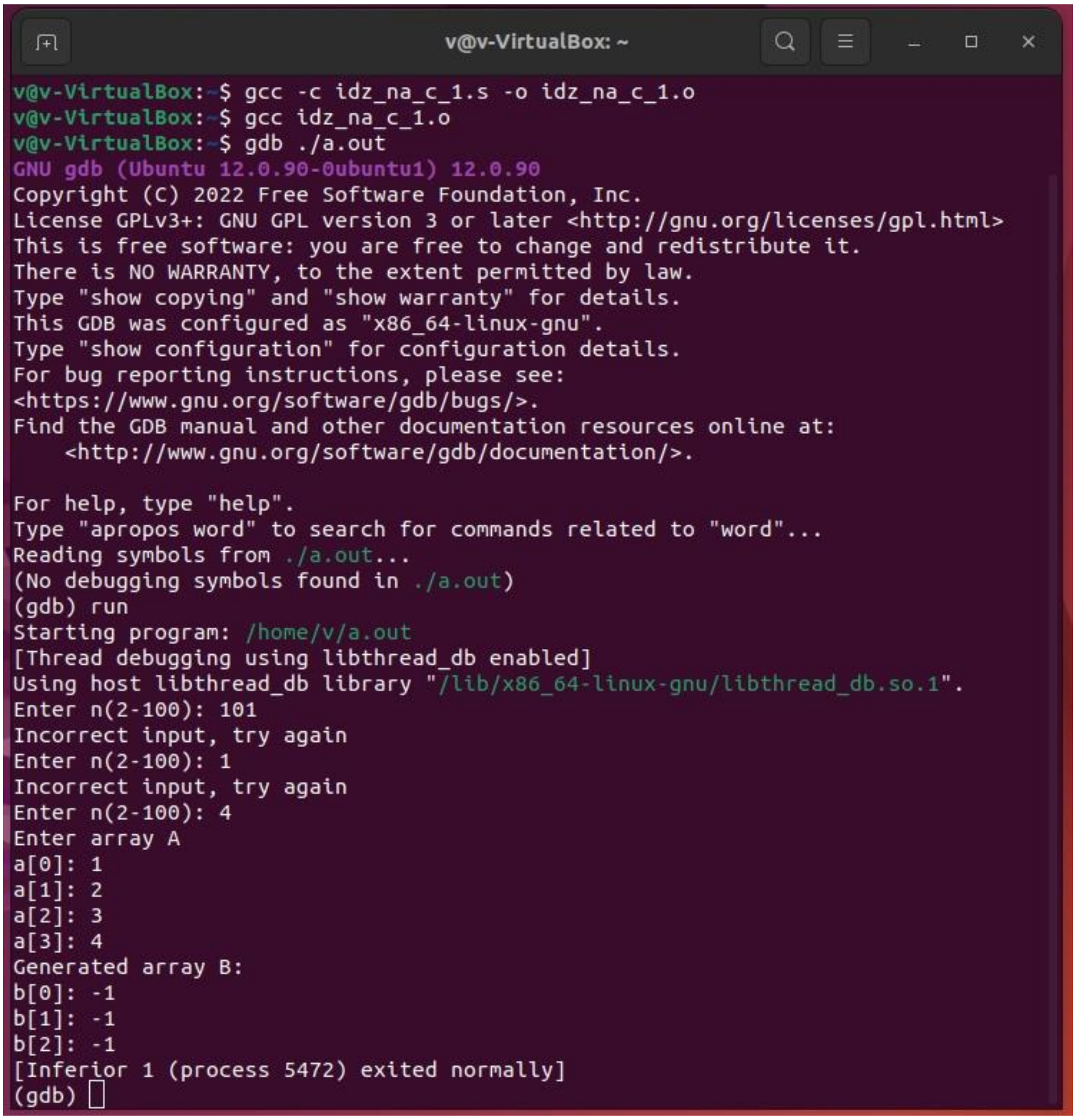
```
subq    $832, %rsp
movq    %fs:40, %rax
movq    %rax, -8(%rbp)
xorl    %eax, %eax
movl    $1, -820(%rbp)          #int ek = 1;
jmp     .L2                     #Переход к блоку L2
.L5:
leaq    .LC0(%rip), %rax        #printf("Enter n(2-100): ")
movq    %rax, %rdi
movl    $0, %eax
call    printf@PLT
leaq    -832(%rbp), %rax        #Считывание n
movq    %rax, %rsi
leaq    .LC1(%rip), %rax
movq    %rax, %rdi
movl    $0, %eax
call    __isoc99_scanf@PLT
movl    -832(%rbp), %eax        #Скопировать содержимое -832 из rbp в eax
cmpl    $1, %eax                #Если <= 1,то переход к блоку L3
jle     .L3
movl    -832(%rbp), %eax        #Скопировать содержимое -832 из rbp в eax
cmpl    $100, %eax              #или >100, то переход к блоку L3
jg      .L3
movl    $0, -820(%rbp)          #Записать 0 в -832 из rbp (ek = 0);
jmp     .L4                     #Переход к блоку L4
.L3:
leaq    .LC2(%rip), %rax        #printf("Incorrect input, try again\n")
movq    %rax, %rdi
call    puts@PLT
.L2:
cmpl    $1, -820(%rbp)  #while (ek == 1) переход к блоку L5
```

```
je      .L5
.L4:
leaq    .LC3(%rip), %rax        #printf("Enter array A")
movq    %rax, %rdi
call    puts@PLT
movl    $0, -828(%rbp)          #Записать 0 в -828 из rbp (int i = 0  в цикле for)
jmp     .L6                     #Переход к блоку L4
.L7:
movl    -828(%rbp), %eax        #printf("a[%d]: ")
movl    %eax, %esi
leaq    .LC4(%rip), %rax
movq    %rax, %rdi
movl    $0, %eax
call    printf@PLT
leaq    -816(%rbp), %rdx        #Считывание a[i]
movl    -828(%rbp), %eax
cltq
salq    $2, %rax
addq    %rdx, %rax
movq    %rax, %rsi
leaq    .LC1(%rip), %rax
movq    %rax, %rdi
movl    $0, %eax
call    __isoc99_scanf@PLT
addl    $1, -828(%rbp)          #i++
.L6:
movl    -832(%rbp), %eax        #Скопировать содержимое -832 из rbp в eax (n)
cmpl    %eax, -828(%rbp)        #for (i < n), то переход к блоку L7
jl      .L7
leaq    .LC5(%rip), %rax        #printf("Generated array B:")
movq    %rax, %rdi
```

```
call    puts@PLT
movl    $0, -824(%rbp)              #Записать 0 в -824 из rbp (int j = 0  в цикле for)
jmp     .L8                         #Переход к блоку L8
.L9:
movl    -824(%rbp), %eax            #Скопировать содержимое -824 из rbp в eax (j)
cltq
movl    -816(%rbp,%rax,4), %edx #Скопировать a[i] из rbp в edx
movl    -824(%rbp), %eax            #Скопировать содержимое -824 из rbp в eax (j)
addl    $1, %eax                    #i+1
cltq
movl    -816(%rbp,%rax,4), %eax
subl    %eax, %edx                  #a[i]-a[i+1]
movl    -824(%rbp), %eax            #Скопировать содержимое -824 из rbp в eax (j)
cltq
movl    %edx, -416(%rbp,%rax,4) #b[j] = a[i]-a[i+1]
movl    -824(%rbp), %eax            #Скопировать содержимое -824 из rbp в eax (j)
cltq
movl    -416(%rbp,%rax,4), %edx           #printf("b[%d]: %d\n")
movl    -824(%rbp), %eax
movl    %eax, %esi
leaq    .LC6(%rip), %rax
movq    %rax, %rdi
movl    $0, %eax
call    printf@PLT
addl    $1, -824(%rbp)              #j++
.L8:
movl    -832(%rbp), %eax            #Скопировать содержимое -832 из rbp в eax (n)
subl    $1, %eax                    #eax-=1
cmpl    %eax, -824(%rbp)            #Если j < n-1, то переход к блоку L9
jl      .L9
movl    $0, %eax
```

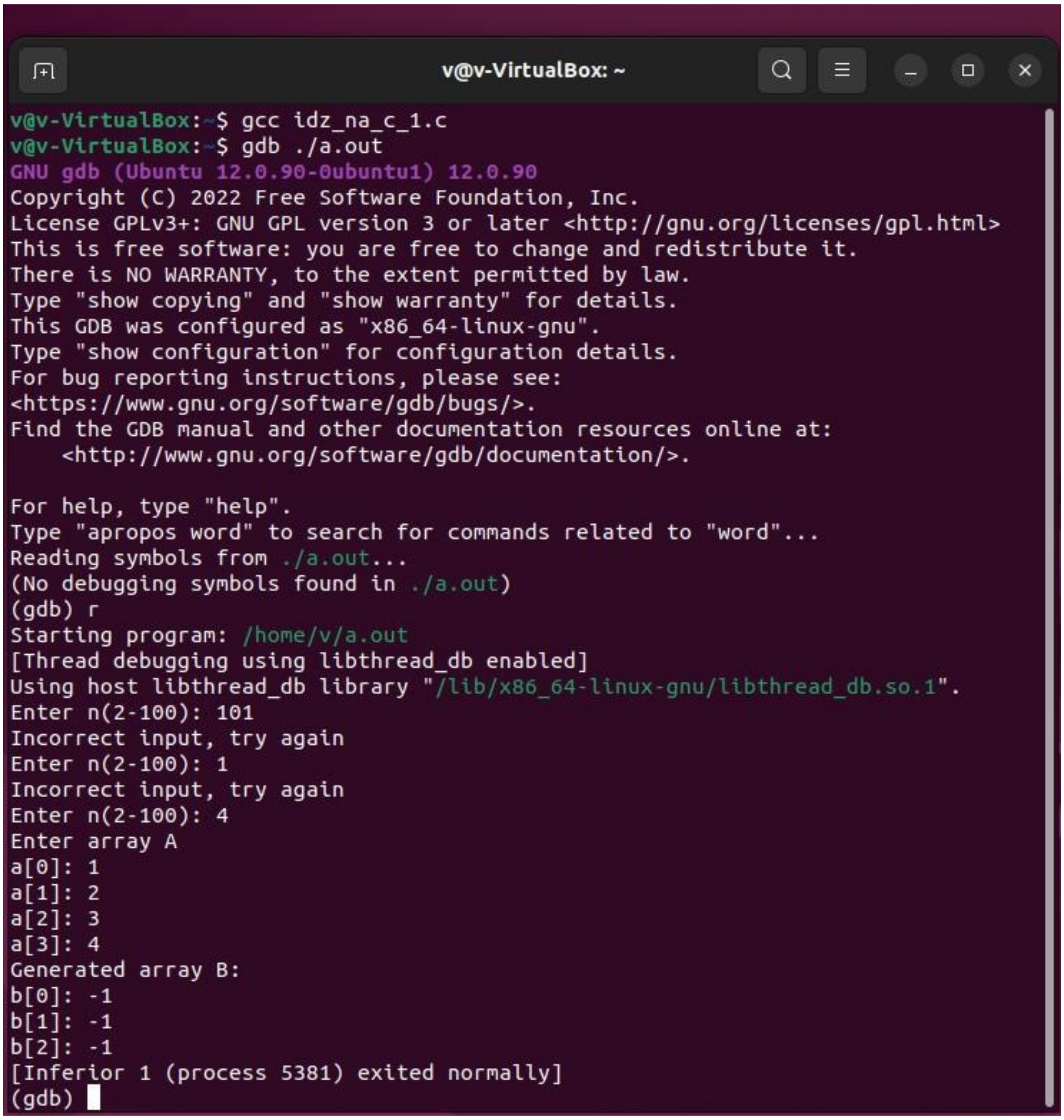
```
movq    -8(%rbp), %rdx
subq    %fs:40, %rdx
je      .L11                        #Если j == n-1, то переход к блоку L11
call    __stack_chk_fail@PLT
.L11:
leave
ret
.size   main, .-main
.ident  "GCC: (Ubuntu 11.2.0-19ubuntu1) 11.2.0"
.section.note.GNU-stack,"",@progbits
.section.note.gnu.property,"a"
.align 8
.long   1f - 0f
.long   4f - 1f
.long   5
0:
.string "GNU"
1:
.align 8
.long   0xc0000002
.long   3f - 2f
2:
.long   0x3
3:
.align 8
4:
```

Сравнительные результаты:



```
v@v-VirtualBox:~$ gcc -c idz_na_c_1.s -o idz_na_c_1.o
v@v-VirtualBox:~$ gcc idz_na_c_1.o
v@v-VirtualBox:~$ gdb ./a.out
GNU gdb (Ubuntu 12.0.90-0ubuntu1) 12.0.90
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./a.out...
(No debugging symbols found in ./a.out)
(gdb) run
Starting program: /home/v/a.out
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Enter n(2-100): 101
Incorrect input, try again
Enter n(2-100): 1
Incorrect input, try again
Enter n(2-100): 4
Enter array A
a[0]: 1
a[1]: 2
a[2]: 3
a[3]: 4
Generated array B:
b[0]: -1
b[1]: -1
b[2]: -1
[Inferior 1 (process 5472) exited normally]
(gdb) 
```



v@v-VirtualBox: ~

```
v@v-VirtualBox:~$ gcc idz_na_c_1.c
v@v-VirtualBox:~$ gdb ./a.out
GNU gdb (Ubuntu 12.0.90-0ubuntu1) 12.0.90
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./a.out...
(No debugging symbols found in ./a.out)
(gdb) r
Starting program: /home/v/a.out
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Enter n(2-100): 101
Incorrect input, try again
Enter n(2-100): 1
Incorrect input, try again
Enter n(2-100): 4
Enter array A
a[0]: 1
a[1]: 2
a[2]: 3
a[3]: 4
Generated array B:
b[0]: -1
b[1]: -1
b[2]: -1
[Inferior 1 (process 5381) exited normally]
(gdb) 
```