Отчёт ИДЗ-2 (ВАРИАНТ 33)

Ставцев М.А.

БПИ-217

4 балла

- Решением на C являются следующие файлы:
  - *main.c* – содержит основную функции *main*, *contains*- для проверки наличия символа в строке и *timespecDiff* для подсчёта затраченного времени; **Важно:** запуск программы всегда необходимо осуществлять с 2 параметрами командной строки: seed и file name, в независимости от планируемого типа ввода данных. Если не будет 2 параметров, то приложение закончит свою работу с кодом ошибки 3.
  - *getData.c* – содержит функцию *getData*, которая считывает две строки в зависимости от переданных в неё параметров (консольный ввод, ввод из файла, случайная генерация);
  - *getRandomString.c* – содержит функцию *getRandomString*, которая генерирует строку символов длиной от 0 до 100 символов. **Важно**: генерация происходи только для ASCII символов от 33 от 126, то есть от символа ‘!’ до символа ‘~’, данное решения принято, чтобы можно избежать генерации символов «начала заголовка» - SOH, «начала текста» - STX и так далее. Генерация ASCII символов от 33 до 126 помогает отследить результат генерации, который выводится в файл *random_gen.txt*;
  - *header.h* – для подключения необходимых библиотек
  - Вывод результирующий строки зависит от типа ввода, консольный или случайная генерация – вывод в консоль, файловый – вывод в файл *output.txt*
- Получение ассемблерных файлов программы, а также получение исполняемых .exe файлов показано далее на скриншотах:

```
mastavtsev@mastavtsev-VirtualBox:~$ gcc -masm=intel \
-fno-asynchronous-unwind-tables \
-fno-jump-tables \
-fno-stack-protector \
-fno-exceptions \
./main.c \
-S -o ./main.s
mastavtsev@mastavtsev-VirtualBox:~$ gcc -masm=intel \
-fno-asynchronous-unwind-tables \
-fno-jump-tables \
-fno-stack-protector \
-fno-exceptions \
./getRandomString.c \
-S -o ./getRandomString.s
mastavtsev@mastavtsev-VirtualBox:~$ gcc -masm=intel \
-fno-asynchronous-unwind-tables \
-fno-jump-tables \
-fno-stack-protector \
-fno-exceptions \
```

Дизассемблирование исходных файлов с ключами компиляции

```
mastavtsev@mastavtsev-VirtualBox:~$ gcc ./main.c -c -o main.o
mastavtsev@mastavtsev-VirtualBox:~$ gcc ./getData.c -c -o getData.o
mastavtsev@mastavtsev-VirtualBox:~$ gcc ./getRandomString.c -c -o getRandomStri
ng.o
mastavtsev@mastavtsev-VirtualBox:~$ gcc ./getData.o ./getRandomString.o main.o
-o foo-C.exe
```

Получение объектных файлов из C - версии программы и получение исполняемого .exe файла

```
mastavtsev@mastavtsev-VirtualBox:~$ gcc ./main.s -c -o main.o
mastavtsev@mastavtsev-VirtualBox:~$ gcc ./getData.s -c -o getData.o
mastavtsev@mastavtsev-VirtualBox:~$ gcc ./getRandomString.s -c -o getRandomStri
ng.o
mastavtsev@mastavtsev-VirtualBox:~$ gcc ./getData.s ./getRandomString.s main.s
-o foo-S.exe
```

Получение объектных файлов из Assembly - версии программы и получение исполняемого .exe файла

- Ассемблерная и C программы одинаково работают на одинаковом тесте, вводимом через консоль. Более полное тестовое покрытие будет продемонстрировано в описании *пунктов 5, 6, 7*:

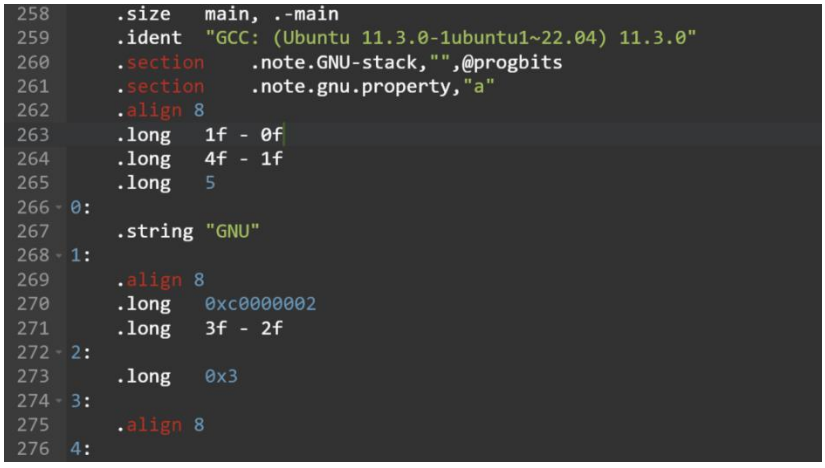
```
mastavtsev@mastavtsev-VirtualBox:~$ ./foo-C.exe 1234 input.txt
Type in the console the type of input you want:
1 - console (output in console)
2 - file input (output in output.txt)
3 - random input (output in console)
1
Enter in console two strings. Divide them by the symbol ';'. The enter process
ends with double Ctrl + D press.
string1_abc;string2_def
RESULT:   string_
Elapsed: 2216 ns
mastavtsev@mastavtsev-VirtualBox:~$ ./foo-S.exe 1234 input.txt
Type in the console the type of input you want:
1 - console (output in console)
2 - file input (output in output.txt)
3 - random input (output in console)
1
Enter in console two strings. Divide them by the symbol ';'. The enter process
ends with double Ctrl + D press.
string1_abc;string2_def
RESULT:   string_
Elapsed: 2323 ns
```

5, 6, 7 баллов

- В данной программе все функции принимают определённые параметры. Сигнатуры функций представлены на данном скриншоте:

```
int main(int argc, char **argv);

extern void getData(int* type, char* str1, char* str2, int* len1,
                    int* argc, int* len2, char **argv);

void getRandomString(char* str1, char* str2, int* len1, int* len2, char* arg);

long timespecDiff(struct timespec start, struct timespec stop);

int contains(char *str, int len, char ch);
```

- Использование локальных переменных – во всех представленных методах используются локальные переменные, например:
  - main
    - int i, n, k – для итерации по строкам
    - char res1[1000], res2[1000] – для хранения промежуточного и итогового вида результирующей строки
  - getData
    - FILE* input – поток ввода
    - int ch – для считывания символов из потока ввода
    - int flag – логическая переменная, сигнализирующая о том, что при вводе был получен символ “;”, что означает конец ввода первой строк и начался ввода второй.
  - contains
    - int flag – логическая переменная, сигнализирующая о нахождении необходимого символа
  - timespecDiff
    - struct timespec ret и long result – используются для подсчёта общего времени, затраченного на сравнение строк и получение результирующей
  - getRandomString
    - int seed – для генерации случайных символов
    - int n1, n2 – для получения размеров строк, которые пользователь вводит в консоль
- Далее будут описаны модификации, которые были совершены в процессе рефакторинга ассемблерного кода, что покрывает пункты на 5, 6 и 7 баллов.
  - main.s (функция main)
    - Удаление метаинформации из конца файла



        - Заменим использование стека DWORD PTR -4[rbp] для переменной для итераций i на использование регистра r15
    - main.s (функция contains)
        - Заменим использование стека DWORD PTR -4[rbp] для переменной flag на использование регистра r14
        - Заменим использование стека DWORD PTR -8[rbp] для переменной для итераций i на использование регистра r13d
    - getData.s
        - Убираем метаинформацию

```
    .size   getData, .-getData
    .ident  "GCC: (Ubuntu 11.3.0-1ubuntu1~22.04) 11.3.0"
    .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:
```

    - getRandomString.s
        - Также убираем мета информацию

- Подробное описание работы ассемблерной программы содержится в комментариях модифицированных файлов.
- Замеры времени и сравнения результатов вывода:
  Параметры командной строки: 879 input.txt
  1. Консольный ввод: str1;str2
     C code:

```
RESULT:    str
Elapsed: 515 ns
```

     Assembly not modified:

```
RESULT:    str
Elapsed: 733 ns
```

Assembly modified:

```
RESULT:    str
Elapsed: 1044 ns
```

2. Ввод из файла input.txt: string1f;string_ssfadd

C code:

```
stringf
The resulting string is in the output.txt file.
Elapsed: 1125 ns
```

Assembly not modified:

```
stringf
The resulting string is in the output.txt file.
Elapsed: 758 ns
```

Assembly modified:

```
stringf
The resulting string is in the output.txt file.
Elapsed: 621 ns
```

3. Случайный ввод (**<u>8 баллов</u>**)

Длина слов 25 и 75. Из-за псевдо-генерации и одинакового seed мы должны получать одинаковые «случайные» слова

C code:

```
U|YGp@%n%jy;m>T^7Yjrv#[=D
]'@;9?,6xl(Sp1Wv+rdba^5;I(Le}h*u+dKC$Vr7CyjOF\FjiENL$b"L%NL#QV3[U^9X5-+rAtB
RESULT:    Up@%jy;^7rv#[
Elapsed: 4324 ns
```

Assembly not modified:

```
U|YGp@%n%jy;m>T^7Yjrv#[=D
]'@;9?,6xl(Sp1Wv+rdba^5;I(Le}h*u+dKC$Vr7CyjOF\FjiENL$b"L%NL#QV3[U^9X5-+rAtB
RESULT:    Up@%jy;^7rv#[
Elapsed: 4112 ns
```

Assembly modified:

```
U|YGp@%n%jy;m>T^7Yjrv#[=D
]'@;9?,6xl(Sp1Wv+rdba^5;I(Le}h*u+dKC$Vr7CyjOF\FjiENL$b"L%NL#QV3[U^9X5-+rAtB
RESULT:    Up@%jy;^7rv#[
Elapsed: 2471 ns
```

4. Пустые строки, консольный ввод.

   Проверка на то, что программы не падают при пустых строках на входе.

   C code:

```
RESULT:
Elapsed: 170 ns
```

   Assembly not modified:

```
RESULT:
Elapsed: 92 ns
```

   Assembly modified:

```
RESULT:
Elapsed: 288 ns
```

В гитхабе прилагается 9 файлов:

1. 3 файла с C-кодом (main.c getData.c getRandomString.c)
2. 3 файла с Assembly не изменённым и некомментированным кодом (main.s getData.s getRandomString.s)
3. 3 файла с Assembly изменённым и откомментированным кодом (main - modified.s getData - modified.s getRandomString - modified.s)