1. Введение в Си и разбор задач

Hello, world!

Балл за задачу: 1

Напишите “Hello, world!” на Си.

Примерные шаги решения описаны в презентации. Вам необходимо выложить на любой файлообменник файл с исходным кодом и скриншот с выводом запущенной программы.

2. Стайлгайд

Обратите внимание, что начиная с этого домашнего задания Вы должны следовать стилю кодирования WebKit!

“Быстрое” вычисление многочлена

Балл за задачу: 1

Написать программу, считающую значение формулы $x^4 + x^3 + x^2 + x + 1$ за два умножения.

Неполное частное

Балл за задачу: 1

Написать программу нахождения неполного частного от деления a на b (целые числа), используя только операции сложения, вычитания и умножения.

Переворот массива

Балл за задачу: 1

Дан массив целых чисел $x[1] \dots x[m+n]$, рассматриваемый как соединение двух его отрезков: начала $x[1] \dots x[m]$ длины m и конца $x[m+1] \dots x[m+n]$ длины n. Не используя дополнительных массивов, переставить начало и конец.

Счастливые билетики

Балл за задачу: 1

Посчитать число “счастливых билетов” (билет считается “счастливым”, если сумма первых трёх цифр его номера равна сумме трёх последних), подсчётом числа билетов с заданной суммой трёх цифр.

3. Процесс компиляции

Простой баланс скобок

Балл за задачу: 1

Написать программу проверки баланса скобок в исходной строке (т.е. число открывающих скобок равно числу закрывающих и выполняется правило вложенности скобок). Считайте, что баланс проверяется для одного типа скобок, а строка может содержать произвольные символы.

Поиск подстроки

Балл за задачу: 1

Заданы две строки: S и S1. Найти количество вхождений S1 в S как подстроки.

Количество нулевых элементов в массиве

Балл за задачу: 1

Написать программу, считающую количество нулевых элементов в массиве.

4. Библиотеки и оптимизации

Начиная с данной домашки, все решения должны сдаваться с помощью Pull Request’ов!

“Оптимальная” сортировка

Балл за задачу: 5

Реализовать приложение, сортирующее поступающий в стандартный поток ввода набор целых чисел (не более 100 штук, это гарантируется). Числа разделены пробелами, последним символом является перенос строки.

Процедура сортировки должна быть реализована на языке ассемблера *достаточно оптимальным образом*. Кодом возврата приложения является количество элементов, участвовавших в сортировке и изменивших свою позицию.

Дополнительные соглашения:

- Чтение поступающих на вход сортируемых элементов необходимо производить с помощью команды `scanf(...)`.
- Файлы с кодом приложения должны иметь расширение `.c`, файл с реализацией сортировки на ассемблере должен иметь расширение `.s`.

Не забудьте приложить инструкцию по сборке!

5. Стек и очередь

Все задачи решаются с помощью стека — его надо реализовать единожды в отдельном модуле, и использовать во всех этих задачах. Чтобы каждую задачу можно было сдавать в отдельной ветке, надо сначала сделать ветку для модуля “стек”, реализовать там стек, а затем уже от неё отвести три ветки для конкретных задач. При этом правки к самому стеку надо делать в ветке для стека, а потом вливать изменения из неё в ветки с задачами. (это довольно типичный процесс разработки, когда одна фича зависит от другой) При этом пуллреквест из ветки со стеком открывать не надо.

Комментарии ко всем функциям из заголовочного файла обязательны.

Продвинутый баланс скобок

Балл за задачу: 6

Написать программу проверки баланса скобок в строке, скобки могут быть трёх видов: (), [], {}. Скобочная последовательность вида ({)} считается некорректной, ({}) — корректной.

Сортировочная станция

Балл за задачу: 6

Написать программу, преобразующую выражение из инфиксной формы в постфиксную. В выражении могут быть знаки +, -, *, /, скобки и цифры.

Пример: (1 + 1) * 2 должно преобразовываться в 1 1 + 2 *.

Алгоритм перевода предлагается найти самостоятельно (алгоритм “сортировочной станции” Э. Дейкстры).

6. Списки

Сортированный список

Балл за задачу: 6

Написать программу, которая в диалоговом режиме позволяет осуществлять следующие операции:

- 0 – выйти
- 1 – добавить значение в сортированный список
- 2 – удалить значение из списка
- 3 – распечатать список

Все операции должны сохранять сортированность. Начинаем с пустого списка.

Считалочка

Балл за задачу: 6

Отряд из 41-го сикария, защищавший галилейскую крепость Массада, не пожелал сдаваться в плен блокировавшим его превосходящим силам римлян. Сикарии стали в круг и договорились, что каждые два воина будут убивать третьего, пока не погибнут все. Самоубийство — тяжкий грех, но тот, кто в конце концов останется последним, должен будет

его совершить. Иосиф Флавий, командовавший этим отрядом, якобы быстро рассчитал, где нужно стать ему и его другу, чтобы остаться последними, но не для того, чтобы убить друг друга, а чтобы сдать крепость римлянам. В нашем случае участвует n воинов и убивают каждого m-го. Требуется определить номер k начальной позиции воина, который должен будет остаться последним. Считать с помощью циклического списка.

7. Контрольная работа № 1

Правила игры:

- Контрольная проверяется один раз. Каждая задача стоит 10 баллов.
- Все задания сдаются одним Pull Request. Ссылка на PR выкладывается на HwProj.
- За игнорирование хороших практик, в т.ч. работы с Git, будем снимать баллы.
- Писать код и использовать Git с помощью друзей и/или LLM — запрещено.
- Код можно сдавать не через Git, штраф — 20% баллов.

Вариант 1

Фибоначчи

Балл за задачу: 10

Посчитать сумму всех чётных чисел Фибоначчи, не превосходящих миллиона.

Палиндром

Балл за задачу: 10

Проверить, является ли строка палиндромом — то есть, читается ли она одинаково в обоих направлениях. Заглавные и строчные буквы считаются разными, пробелы должны игнорироваться. Пример палиндрома: “я иду с мечем судия”. Строку можно задать как константу в коде, можно сделать ввод с клавиатуры. Для определения длины строки можно использовать функцию `strlen` из `string.h`.

Сортировка

Балл за задачу: 10

Реализовать гномью сортировку.

“Гномья сортировка основана на технике, используемой обычным голландским садовым гномом (нидерл. tuinkabouter). Это метод, которым садовый гном сортирует линию цветочных горшков. По существу он смотрит на текущий и предыдущий садовые горшки: если они в правильном порядке, он шагает на один горшок вперёд, иначе он меняет их местами и шагает на один горшок назад. Граничные условия: если нет предыдущего горшка, он шагает вперёд; если нет следующего горшка, он закончил.” (c) https://ru.wikipedia.org/wiki/Гномья_сортировка.

Вариант 2

Наибольшая сумма цифр

Балл за задачу: 10

В массиве целых чисел найти число, сумма цифр которого была бы наибольшей. Если таких чисел несколько, вывести на экран все эти числа.

Обезьянья сортировка

Балл за задачу: 10

Реализовать “Обезьянью сортировку” — сортировку, которая случайно перемешивает массив до тех пор, пока он не окажется отсортирован.

Другой палиндром

Балл за задачу: 10

Напечатать на экран все из диапазона [1;n], которые являются палиндромами в двоичной записи. Выводить в десятичной системе счисления.

Вариант 3

Цифра Фибоначчи

Балл за задачу: 10

Посчитать n-ую цифру последовательности Фибоначчи, записанную в строку без пробелов.

Сортировка выбором

Балл за задачу: 10

Реализовать сортировку выбором.

https://ru.wikipedia.org/wiki/%D0%A1%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0

Другая наибольшая сумма цифр

Балл за задачу: 10

В массиве целых чисел найти число, сумма цифр в двоичном представлении которого была бы наибольшей. Если таких чисел несколько, вывести на экран все эти числа.

8. Абстрактные типы данных

Нет домашнего задания.

9. Системы сборки

CMake для домашки № 5

Балл за задачу: 4

Перенести на CMake домашнее задание к 5 занятию (Продвинутый баланс скобок и Сортировочная станция).

Флаги оптимизаций указывать не нужно. А вот дополнительные предупреждения включить нужно.

CMake для остальных домашек

Дополнительные баллы за перенос остальных домашних заданий на CMake.

1 домашка = 1 PR = 1 балл.

Дедлайна к этим задачам нет.

Применимо к домашкам 1-4, 6

10. Тестирование и отладка

Тесты к Сортированному списку

Балл за задачу: 4

Написать тесты к домашке “Сортированный список”.

Тесты должны запускаться по флагу --test, переданному исполняемому файлу.

Также можно настроить запуск тестов через ctest (см. https://coderefinery.github.io/cmake-workshop/testing/)

Не забывайте проверять граничные случаи. Предпочтительно сдавать в отдельной ветке.

11. Линтеры и санитайзеры

clang-format в CI

Балл за задачу: 2

Вспомните про запуск инструментов в CI из курса по Python.

По инструкции отсюда: https://github.com/WoWaster/spbu-c-ci-example, настройте себе clang-format (раздел “В случае отсутствия CMake”) в CI.

12. Внутреннее представление данных

При выполнении домашнего задания не забывайте использовать CMake, а также писать тесты к своему коду.

Двоичное представление

Балл за задачу: 4

Ввести два числа, перевести в двоичное представление в дополнительном коде и напечатать, сложить в столбик в двоичном представлении, вывести сумму, перевести в десятичное, вывести сумму в десятичном виде.

Double под микроскопом

Балл за задачу: 4

По содержимому памяти вывести значение типа double в экспоненциальной форме: $sm \cdot 2^p$, где s — знак мантиссы, m — мантисса, p — порядок числа.

Примеры:

```
Enter a number: -2.5
Result: -1.25*2^1

Enter a number: 12312.323
Result: +1.5029691162109375384*2^13
```

Обратите внимание, что мантисса выводится в форме нормализованного числа.

Во время выполнения задания запрещено пользоваться функциями, которые самостоятельно извлекают необходимые значения (например, frexp). Вы можете конвертировать double в удобный вам АТД и уже взаимодействовать с ним. Можно конвертировать в char, bool или любой другой удобный и написанный вами формат.

Полезно будет вспомнить в каком порядке лежат данные в памяти большинства современных компьютеров!

Подсказка:

В C есть ключевое слово union, которое позволяет создавать объединенные по памяти структуры. Другими словами, поля объявленной такой структурой будут иметь общую память:

```
typedef union Number {
   double value;
   char binaryForm[8];
} Number;

> sizeof(Number);
8
```

В случае struct размер был бы 16 байт (8 байт на double и 8*1 байт на char). Имея такую структуру вы можете сначала считать value, а затем работать с binaryForm. Его можно честно сконвертировать в bool, а можно просто написать функцию getBit для извлечения i-го бита.

Подсказка для сильных духом:

В качестве второго поля union можно использовать не массив char, а структуру с bit-field, соответствующими размеру знака, мантиссы и экспоненты.

13. Битовые операции

Домашнего задания нет.