Практикум №2 по АиАЯ

Реализуйте на языке C++ контейнер, реализующий структуру данных «Стек». Для апробации контейнера реализуйте эмулятор виртуального процессора.

(А) Требования этапа «Прототип стека»:

1. Библиотека определяет класс «Стек». Для стека определены операции:
- Создание/удаление: конструктор копирования/перемещения, присваивание копированием/перемещением, деструктор (см. Правило Пяти);
- Модификация стека (push/pop): каждая из операций определена для копирующей семантики и для семантики перемещения;
- Чтение элементов (top): чтение верхнего элемента стека.
- За прототип интерфейса можно взять std::stack.
2. Решение доступно в формате публичного репозитория.
3. Сборка осуществляется с помощью системы сборки Make или CMake.
4. Реализован тестовый набор:
- Каждая функция библиотеки покрыта хотя бы одним тестом;
- Результаты тестов наглядные и помогают в отладке.
5. Стек корректно работает с элементами любого типа (см. Шаблоны).

(Б) Требования этапа «Прототип эмулятора»

6. Реализован эмулятор виртуального процессора заданной архитектуры:
- Эмулятор принимает путь к файлу программы для эмулируемого процессора в качестве аргумента консоли;
- Программа для эмулируемого процессора состоит из команд, разделённых пробелами и символами переноса строки.
7. Эмулируемый CPU реализует команды №1-№12. Для эмулируемого CPU написана программа, выводящая первые 40 чисел Фибоначчи.

(В) Требования этапа «Доработка эмулятора»:

8. Эмулируемый CPU реализует команды №1-№19. Для эмулируемого CPU написана программа, выводящая факториал числа, введённого с консоли.
9. Эмулируемый CPU реализует команды №1-№21. Для эмулируемого CPU написана программа, рекурсивно вычисляющая факториал числа.
10. Стадия преобразования программы в байт-код и стадия исполнения разделены, и выполняются отдельными командами консоли. Байт-код сохраняется в промежуточный исполняемый файл.

Сроки реализации:

Этап	Выполненные требования	Срок сдачи
Прототип стека	(А)	29 февраля
Прототип эмулятора	(А),(Б)	14 марта
Доработка эмулятора	(А),(Б),(В)	21 марта

Набор команд виртуального CPU

№	Команда	Семантика команды
1	BEGIN	Перевод процессора в режим исполнения команд.
2	END	Остановка выполнения команд.
3	PUSH *value0*	Добавление на вершину стека числа *value0*.
4	POP	Удаление числа из вершины стека.
5	PUSHR *reg0*	Чтение регистра *reg0* и добавление его на вершину стека.
6	POPR *reg0*	Удаление числа из вершины стека и его запись в регистр *reg0*.
7	ADD	Сложение двух верхних значений из стека с сохранением результата на вершине стека.
8	SUB	Вычитание двух верхних значений из стека с сохранением результата на вершине стека.
9	MUL	Умножение двух верхних значений из стека с сохранением результата на вершине стека.
10	DIV	Деление двух верхних значений из стека с сохранением результата на вершине стека.
11	OUT	Удаление числа из вершины стека с выводом в консоль.
12	IN	Считывание из консоли числа с сохранением на вершине стека.
13	JMP *label*	*Безусловный переход* на метку *label*.
14	JEQ *label*	Условный переход на метку *label*, если верхние два числа на стеке равны.
15	JNE *label*	Условный переход на метку *label*, если верхние два числа на стеке не равны.
16	JA *label*	Условный переход на метку *label*, если верхнее число на стеке больше следующего за ним.
17	JAE *label*	Условный переход на метку *label*, если верхнее число на стеке больше следующего за ним или равно ему.
18	JB *label*	Условный переход на метку *label*, если верхнее число на стеке меньше следующего за ним.
19	JBE *label*	Условный переход на метку *label*, если верхнее число на стеке меньше следующего за ним или равно ему.
20	CALL *label*	Вызов функции, тело которой располагается по метке *label*.
21	RET	Возврат из функции.

Архитектура виртуального CPU