**Защита лабораторной №1**

1. Абстрактная машина является потомком принстонской архитектуры (Фон Неймана) и состоит из 3 частей:

* Центральный процессор – калькулятор для данных, управляет загрузкой/выгрузкой данных памяти и оборудованием (внутри ЭВМ думает только процессор)
* ОЗУ – оперативное запоминающее устройство (RAM – random access memory) – хранит данные и код. Это одномерный массив ячеек, к каждой из которых можно обратиться по адресу.

Минимальная единица адресации – минимальная разница между двумя разными адресами (в нашей машине = 1 байт)

* Система ввода-вывода (IO) – общение с внешним оборудованием (включая жесткий диск – потому что внутри жесткого диска есть свой процессор)

1. Машинное слово – сообщения фиксированной длины, с помощью которых компьютер общается с составными частями.

Величина машинного слова на практике – 4-8 байт

1. Транслятор – программа-переводчик с языка высокого уровня на язык машины.
2. Ключевое слово – слово, зарезервированное транслятором для собственных нужд (if, for, while).
3. Переменную можно назвать IF, If, нельзя – if
4. Точка входа – строка, с которой начинается исполнение программы (В СИ она единственна)

Точка выхода – строка, которой заканчивается выполнение программы (В СИ может быть несколько)

1. Препроцессинг – комплекс действий над текстом на языке высокого уровня перед основной трансляцией. Транслятор не видит решетку.
2. Команды препроцессора: include – вставляет вместо директивы содержимое файла, define с 2 псевдоаргументами – заменяет одно на второе (константы, макроопределения). Define не работает внутри строчных литералов.
3. В стандарте си в библиотеке нет ни одной математической константы.
4. Можно, вопрос в удобстве написания кода (не писать громадные формулы)
5. Стандарт определяет язык программирования. Компилятор может не поддерживать стандарт.
6. Стандартная библиотека – поставляется вместе с языком программирования и включает в себя некий необходимый набор функционала - набор модулей, констант, глобальных переменных, шаблонов, макросов, функций и процедур, доступных для вызова из любой программы.
7. Stdio.h – заголовочный файл ввода-вывода стандартной библиотеки СИ, содержащий определения макросов, констант, объявления функций и типов данных для ввода и вывода

Stdlib.h - заголовочный файл стандартной библиотеки СИ, который содержит в себе функции, занимающиеся выделением памяти, контролем процесса выполнения программы, преобразованием типов

Math.h - заголовочный файл стандартной библиотеки языка программ,

Содержащий математические ф-и (нет констант, есть ф-и)

1. Тип данных – множество значений и операций над ними.

Простой тип данных – неделимый тип данных - никакая сущность данного типа не может включать в себя сущность простого типа. Переменные простого типа могут содержать только одно значение, а составного типа – несколько однотипных или разнотипных значений.

В стандарте существуют два правила, которых должны придерживаться все компиляторы: • пункт 5.2.4.2 определяет минимальный диапазон значений (limits.h) • пункт 6.3.1.1 определяет относительные размеры типов sizeof(short int) <= sizeof(int) <= sizeof(long int) <= sizeof(long long int)

1. Short int, int, long int, long long, unsigned int, unsigned short int, unsigned long int, char + беззнаковые (unsigned, size\_t);

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Тип | Размер | Операции | Спец | Дополнительно |
| Short | >= 2байта | Множество арифметических  операций | %sd | всегда приводится к машинному слову (исходя из того, что длина машииного слова 4-8 байт, а short >=2 |
| Long | >= 4 байта | -|| - | %ld |  |
| Long long | >= 8 байт | -||- | %lld |  |
| Int | Sizeof (short) <= sizeof (int) <= sizeof( long) | -||- | %d | Нельзя говорить, что инт = 4 байта! |
| Char | Ровно 1 байт | Множество арифметических  Операций + операции с символами | %c | это минимальная единица буфера. Так, например, char в 100 элементов – буфер в 100 байт |



Для каждой конкретной машины диапазон можно определить с помощью заголовочного файла limits.h

1. Переполнение знаковых – неопределенное поведение (undefined behavior – ‘хуже, чем ошибка, потому что она прячется’)

Переполнение беззнаковых происходит по кольцу (с нуля) – эти числа замкнуты и относительно умножения, сложения и вычитания (в отличие от знаковых)

Все целые типы замкнуты относительно деления нацело и взятия остатка

1. Одинарная точность – float (floating point of number) – 4 байта – 6-8 значащих разрядов (32-бит – мантисса = 24 бита)

Двойная точность – double (double precision floating point number)– 8 байт – 12-15 значащих разрядов (64-бит – мантисса = 52 бита)

1. Нельзя сравнивать два чпт через ==
2. Необходимо вводить eps = 1e-8 (например, как у нас), и сравнивать

|x1 – x2|<=eps

1. 1 × 10 ^ (−12) + 1 × 10 ^ 12 = 1 × 10 ^ 12
2. 1 × 10 ^ (−12) - 1 × 10 ^ 12 = - 1 × 10 ^ 12
3. 1 × 10 ^ (−16) + 1 × 10 ^ (-17) = 1,1 × 10 ^ (-16)
4. Чегооооооо? Мой ответ, насколько я поняла, речь о количестве знаков после запятой?
5. y = 3.1234567891234 + 2.1f = 5.223457
6. а) определить, целые (например, кол-во чего-то) или чпт

б) если экономия 4 байт не играет роли, то выбираем double, иначе – float

в) если нужна двойная ширина диапазона спереди – беззнаковое, иначе – знаковое

г) если строгое соответствие размеру – фиксированное целое (size\_t)

д) если переменная – флажок, код возврата или она максимально приближена к машинной – int

e) большой диапазон -long long

ё) буферная переменная для массива произвольной длины – char

ж) short – никогда (машинное слово все равно большего размера)

1. Приведение типа – изменение типа выражения в соответствии с заранее заданным набором правил транслятора.

Явное приведение типа – приведение типа по запросу программиста.

Неявное приведение – приведение, вызванное невозможностью осуществить действие без приведения.

1. Выражение 4 <= 3.9 будет сначала приведено к дубле, т.к. в приоритете у транслятора минимизация потери информации, и вообще-то операция сравнения инт и дабл в си не определена.
2. Си – язык со статической типизацией, поэтому тип переменной менять нельзя, но можно менять тип выражения.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| + | long int | Long long | float | Double | Uint32\_t | Uint64\_t |
| long int | Long int |  |  |  |  |  |
| Long long | ll | ll |  |  |  |  |
| float |  | float | float |  |  |  |
| Double |  |  |  | double |  |  |
| Uint32\_t |  |  |  |  | Uint32\_t |  |
| Uint64\_t |  |  |  |  |  | Uint64\_t |

1. Целое число может служить логическим выражением
2. Всегда ложно число 0 и только оно, остальные целые числа - истина
3. Порядок арифметических вычислений определен
4. a+b+c+d+e+g – левая ассоциативность
5. f(a)+f(b)+f(f(a)) – аргументы ф-й рандомно выполняется, а сложение ф-й упорядоченно
6. Да, строго левоассоциативный. Булевы выражения имеют нестрогий порядок вычисления, называемый вычислением по короткой схеме (short-circuit evaluation), где вычисления прекращаются, как только результат становится однозначно известен
7. Такая конструкция возможна благодаря нестрогой модели вычислений (non-strict evaluation) – аргументы не вычисляются до тех пор, пока их значения не будут использованы в теле функции
8. 27 && 13 && 1 && 12 = ИСТИНА
9. Операторные скобки — скобки или команды, определяющие в языке программирования блок команд, воспринимаемый как единое целое, как одна команда
10. If (выражение)

Оператор 1

Else

Оператор 2

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

if (выражение)

Оператор 1

Else if

Оператор 2

Else if

Оператор 3

…

Else

Оператор

1. Виды циклов: счетные – с известным заранее количеством итераций), с предусловием – возможно, что тело цикла не выполняется ни разу, с постусловием – выполняется хотя бы одна итерация.

For, while, while do

1. Выражение – возвращает значение – (код (часть кода), определяющий значение)

Инструкция — команда – ( это некоторое законченное выражение в программном коде. После инструкции обязательно должен стоять ;)

Конструкции нелегальны.

1. С помощью цикла с предусловием можно написать любой другой.
2. Числовая последовательность – пронумерованный набор элементов числового мн-ва

Элементы последовательности связаны рекуррентной формулой, если, начиная с определенного члена k, каждый элемент зависит от всех предыдущих.

Простая рекуррентная формула – связывает элемент последовательности только с предыдущим:

- мультипликативная – каждый следующий равен предыдущему, умноженному на число

- аддитивная - каждый следующий равен предыдущему, к которому прибавлено число

47.

49. factorial (i) / 20^i

***Дополнительно:***

Принципы архитектуры фон Неймана:

- принцип однородности памяти – команды и данные хранятся в одной и той же памяти и внешне в памяти не различимы

- принцип адресности – структурно основанная память состоит из пронумерованных ячеек и каждая ячейка всегда доступна в произвольный момент по ее адресу

- принцип программного управления - все вычисления, предусмотренные алгоритмом решения задачи, должны быть представлены в виде программы, состоящей из последовательности управляющих слов — команд

В x32 архитектуре слово состоит из 4 байт, в x64 – из 8.

Стандарт языка программирования – документация, определяющая язык программирования, позволяющая разработчикам согласовывать, что означают программы на данном языке.

Компиляция – препроцессинг + трансляция + сборка

Компоновщик – программа, преобразующая набор оттранслированных модулей в единую программу.

Макрос – предопределенная структура, заданная процессору, который поставит ее в нужном месте.

Структура — конструкция большинства [языков программирования](https://dic.academic.ru/dic.nsf/ruwiki/1465), позволяющая содержать в себе набор [переменных](https://dic.academic.ru/dic.nsf/ruwiki/4894) различных типов.

Стандарт чисел с плавающей точкой – IEEE 754

Одинарная точность – float – 32-битная – мантисса = 24 бита; Двойная точность – double – 64-битная – мантисса = 52 бита;

Арифметические операции правоассоциативны, операции отношения (> <) левоассоциативны

Если оба операнда операции “/” являются целочисленными, выполняется целочисленное деление (т.е. дробная часть просто отбрасывается):

1 / 2 === 0 (не 0.5)

В операции “%” оба операнда должны быть целочисленными (в противном случае возникнет ошибка компиляции). Использование 0 в качестве правого операнда операций “/” или “%” приведет к непредсказуемому результату (неопределенное поведение).

Если оба операнда операций “/” и “%” являются целочисленными и один из них отрицательный, результат зависит от используемого стандарта.

В C99 результат деления округляется большую сторону (по направлению к нулю).

-9 / 7 === -1

–9 % 7 === 2

A = 5 + 4 – 9 #выражение

5 + 4 #операция

Выражение всегда возвращает значение и представляет собой конструкцию, состоящую из констант, операций, скобок и функций.

Операция — конструкция в языках программирования, аналогичная по записи математическим операциям, то есть специальный способ записи некоторых действий.

Подпрограмма – именованная часть программы, содержащая описание определённого набора действий.

Функция – это подпрограмма специального вида, которая всегда должна возвращать результат. Вызов функции является, с точки зрения языка программирования, выражением.

Процедура – это независимая именованная часть программы, которую после однократного описания можно многократно вызвать по имени из последующих частей программы для выполнения определенных действий.

Адрес переменной – это ячейка памяти, с которой связана данная переменная. Адрес переменной иногда называют ее левым значением (l-value), потому что именно адрес переменной требуется, когда имя переменной появляется слева от операции присваивания.

Значение переменной – это содержимое ячейки или ячеек памяти, связанных с данной переменной. Значение переменной иногда называется ее правым значением (r-value), потому что именно оно необходимо при использовании переменной, указанной в правой части операции присваивания.

1 байт – минимальная единица, которую обрабатывает компьютер

Операции в си оптимизируются, но только на том уровне, когда это не будет влиять на вычисления. (например, операция удаление мертвого кода – мертвый код – кусок кода, результат вычислений которого не повлияет на общий результат)