1. Почему шестнадцатеричные числа получили широкое распространение в системном и встраиваемом программировании? В чем их преимущество перед использованием двоичных?

Ответ: удобство работы с такими числами, простой перевод в двоичный код, удобство чтения. Каждая шестнадцатеричная цифра содержит в себе 4 двоичных

1. Сколько бит занимает 1 шестнадцатеричная цифра (hex-цифра)?

Ответ: 4 бита

1. Сколько hex-цифр в ширину насчитывает 1 байт, 2 байта, 4 байта, 8 байт?

Ответ: 1, 2, 4, 8

1. Почему не рекомендовано использование целочисленных типов short, int и long во встраиваемых приложениях? Какие целочисленные типы нужно использовать и где объявлены их определения? Назови длину в байтах для каждого типа

Ответ: для int выделяется изначально 4 байта. Чтобы такого избежать, необходимо явно объявлять, сколько бит выделить под число (например: uint8\_t для выделения 8 бит). Использование int во встраиваемых систем опасно, так как имеются разные архитектуры МК (8, 16, 32 разрядные и т.д.)

Необходимо использовать uint8\_t, uint16\_t, uint32\_t (u – беззнаковый). Объявляется такое определение перед простым числом (например: *(uint16\_t)465)*)

uint8\_t – 1 байта (аналог – char)

uint16\_t – 2 байта (аналог – short)

uint32\_t – 4 байта (аналог – int)

1. В чем разница между константной переменной (константой) и константным литералом (литералом)?

Ответ: Константа определяется до функции с помощью const(*const int X = 25*). Константный литерал же объявляется без const (int X = (uint16\_t)228)). Литералом также может быть символ (например: ‘x’) или строка (например: “x”)

1. Каким типом по умолчанию принимается целочисленный литерал, вещественный литерал и символьный литерал (например, 'с' или '1')? Каким образом можно изменить этот тип литерала?

Ответ: Int, float и char

1. В чем разница между 'a' и "a" в Си? В смысле как называется один литерал и другой и в чем их разница размещении в памяти?

Ответ: ‘a’ – символ char (1 байт), “a” – строка char name[] (имеет последний нулевой символ и занимает уже 2 байта). Нулевой символ говорит об окончании строки

1. Как использовать в Си логический (булевый) тип? С какого стандарта Си возможно его применение?

Ответ: добавить библиотеку, которая позволит значениям TRUE и FALSE присваивать единицу и ноль, соответственно (stdbool.h). Переопределяет \_BOOL на bool

bool x = false; (0)

bool y = true; (1)

1. Что обычно понимается под "магическим числом" (magic number)? Почему не рекомендовано их использование и чем обычно заменяют "магическое число" в коде?

Ответ: unsigned int x = 10;

Под переменную x будет выделено 4 беззнаковых байта, но 10 – литерал, который хранится во flash памяти в бинарном виде, и программа не понимает, сколько байт выделить под этот литерал. Под 10 выделится 4 ЗНАКОВЫХ байта. Чтобы сказать, что литерал будет занимать 2 байта и литерал будет считаться беззнаковым пишется *(uint16\_t)10u*

10 в данном случае – magic number. В большом коде можно запутаться в том, откуда это число взять и что значит. Для предотвращения таких случаев подобные числа объявляют в дефайне в начале файла:

#define SOMETHING 10u

При сравнении двух чисел x и y, где x – беззнаковое, а y – без указания, можем получить ситуацию, в которой y по умолчанию будет объявлен как знаковый. При сравнении беззнакового и знакового, беззнаковое станет знаковым (беззнаковые 8 битные – от 0 до 255, а беззнаковые – от -128 до 127

1. Как правильно выполнять сравнение вещественных переменных?

Ответ:

int compare(float x, float y)

{

return (x – y < FLT\_EPSILON) && (x – y < FLT\_EPSILON);

}

int main(void)

{

float x;

float y;

if (compare(x, y)) {

}

return 0;

}

FLT\_EPSILON – некоторое малое число из библиотеки float.h. Если результат x – y и y – x будет меньше, чем это число FLT\_EPSILON, то тогда считается, что числа float равны. Аналогично работает с double

1. Продемонстрируй объявление с инициализацией символа и строки любым значением

Ответ: char a = ‘a’;

char b[] = “bdzvv”;