# РАЗРАБОТКА КРОССПЛАТФОРМЕННЫХ ПРОГРАММНЫХ СИСТЕМ

## Билеты к экзамену

## 2022

## Вариант №1

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| unsigned int | 8 | Нет | 4 |
| char | 1 | Нет | 1 |

Порядок байт от старшего к младшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 unsigned int field1;  
 char field2;  
 char field3[8];  
 unsigned int field4;  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для преобразования структуры data в массив байт. В реализации необходимо учесть особенности платформы. Вывести на экран значения массива байт, как показано в примерах ниже:

**Пример 1**

Функция для заполнения полей структуры data:

void test\_code1(struct data \*d) {  
 d->field1 = 4097912552115920292;  
 d->field2 = 11;  
 d->field3[0] = 176;  
 d->field3[1] = 95;  
 d->field3[2] = 179;  
 d->field3[3] = 255;  
 d->field3[4] = 156;  
 d->field3[5] = 55;  
 d->field3[6] = 140;  
 d->field3[7] = 167;  
 d->field4 = 3551640973236821270;  
}

Результат вывода массива байт на экран:

38 DE B5 C1 25 7B DD A4   
0B B0 5F B3 FF 9C 37 8C   
A7 00 00 00 31 49 F6 A5   
D3 80 DD 16

**Пример 2**

Функция для заполнения полей структуры data:

void test\_code2(struct data \*d) {  
 d->field1 = 1822677561619647631;  
 d->field2 = 227;  
 d->field3[0] = 144;  
 d->field3[1] = 226;  
 d->field3[2] = 211;  
 d->field3[3] = 245;  
 d->field3[4] = 28;  
 d->field3[5] = 186;  
 d->field3[6] = 53;  
 d->field3[7] = 155;  
 d->field4 = 14758400869715079791;  
}

Результат вывода массива байт на экран:

19 4B 73 95 D8 6A 0C 8F   
E3 90 E2 D3 F5 1C BA 35   
9B 00 00 00 CC D0 5F 65   
CC 2E F2 6F

## Вариант №2

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| short | 2 | Да | 4 |
| unsigned long long | 8 | Нет | 4 |
| unsigned short | 2 | Нет | 4 |
| char | 1 | Да | 1 |
| long long | 8 | Да | 4 |

Порядок байт от младшего к старшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 short field1;  
 unsigned long long field2;  
 unsigned short field3[8];  
 char field4;  
 long long field5;  
 long long field6;  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для преобразования структуры data в массив байт. В реализации необходимо учесть особенности платформы. Вывести на экран значения массива байт, как показано в примерах ниже:

**Пример 1**

Функция для заполнения полей структуры data:

void test\_code1(struct data \*d) {  
 d->field1 = 22991;  
 d->field2 = 1013621258870673659;  
 d->field3[0] = 32094;  
 d->field3[1] = 29207;  
 d->field3[2] = 21791;  
 d->field3[3] = 44171;  
 d->field3[4] = 22825;  
 d->field3[5] = 49868;  
 d->field3[6] = 37060;  
 d->field3[7] = 47519;  
 d->field4 = 48;  
 d->field5 = 7229216298569543411;  
 d->field6 = 8299211933111233434;  
}

Результат вывода массива байт на экран:

CF 59 00 00 FB 50 C5 1F   
2A 1B 11 0E 5E 7D 00 00   
17 72 00 00 1F 55 00 00   
8B AC 00 00 29 59 00 00   
CC C2 00 00 C4 90 00 00   
9F B9 30 00 F3 AE B3 E8   
EB 55 53 64 9A 2B C7 34   
48 B9 2C 73

**Пример 2**

Функция для заполнения полей структуры data:

void test\_code2(struct data \*d) {  
 d->field1 = 11055;  
 d->field2 = 7539719808075053352;  
 d->field3[0] = 23373;  
 d->field3[1] = 45361;  
 d->field3[2] = 36968;  
 d->field3[3] = 9421;  
 d->field3[4] = 60161;  
 d->field3[5] = 60769;  
 d->field3[6] = 37974;  
 d->field3[7] = 999;  
 d->field4 = 117;  
 d->field5 = -4991259017130799714;  
 d->field6 = -5422094549228164397;  
}

Результат вывода массива байт на экран:

2F 2B 00 00 28 39 4A E2   
37 77 A2 68 4D 5B 00 00   
31 B1 00 00 68 90 00 00   
CD 24 00 00 01 EB 00 00   
61 ED 00 00 56 94 00 00   
E7 03 75 00 9E 0D F4 65   
5E 7C BB BA D3 6A AC 26   
BC D9 C0 B4

## Вариант №3

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| long long | 8 | Да | 8 |
| char | 1 | Да | 1 |
| unsigned long long | 8 | Нет | 8 |
| unsigned long | 8 | Нет | 1 |

Порядок байт от младшего к старшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 long long field1;  
 char field2;  
 unsigned long long field3;  
 char field4;  
 unsigned long field5[2];  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для разбора массива байт, содержащего структуру data. В реализации необходимо учесть особенности платформы. Вывести на экран значения полей, как показано в примерах ниже.

**Пример 1**

Массив байт, содержащий значения структуры data:

unsigned char test\_data1[] = {  
 0x71, 0x3c, 0xc3, 0x8c, 0x77, 0x38, 0x2e, 0x0a,   
 0x39, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   
 0xc2, 0x47, 0x39, 0x7d, 0xc9, 0xdb, 0xb8, 0xe9,   
 0x65, 0xff, 0xb1, 0xf5, 0xaf, 0x99, 0xe5, 0x3c,   
 0x6c, 0x99, 0x20, 0x24, 0x90, 0x8d, 0x74, 0xfc,   
 0xc2,   
};

Результат вывода значений полей data на экран:

733585875421838449  
57  
16841452464987785154  
101  
7799361102943728127 14050233088846602393

**Пример 2**

Массив байт, содержащий значения структуры data:

unsigned char test\_data2[] = {  
 0x1f, 0x4d, 0xe8, 0xf1, 0x7d, 0xa2, 0x37, 0xa1,   
 0x02, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   
 0xb3, 0xe8, 0x8e, 0x17, 0x5e, 0x01, 0x15, 0xa1,   
 0x70, 0x55, 0x0c, 0x09, 0x00, 0x6e, 0xe7, 0x5a,   
 0x9f, 0xd6, 0xbd, 0x76, 0xe7, 0x1e, 0xb0, 0xcd,   
 0x10,   
};

Результат вывода значений полей data на экран:

-6829811648070922977  
2  
11607185118251116723  
112  
11482744659567512661 1210817521611357654

## Вариант №4

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| char | 1 | Да | 1 |
| unsigned long | 8 | Нет | 8 |
| unsigned short | 4 | Нет | 2 |

Порядок байт от младшего к старшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 char field1[10];  
 unsigned long field2;  
 unsigned short field3;  
 unsigned short field4;  
 char field5;  
 char field6;  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для разбора массива байт, содержащего структуру data. В реализации необходимо учесть особенности платформы. Вывести на экран значения полей, как показано в примерах ниже.

**Пример 1**

Массив байт, содержащий значения структуры data:

unsigned char test\_data1[] = {  
 0x36, 0xa6, 0xfa, 0x98, 0x47, 0xb6, 0xa2, 0x05,   
 0x7c, 0xdf, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   
 0x63, 0x41, 0x22, 0x2e, 0xcf, 0x59, 0x42, 0xd9,   
 0x8a, 0xe6, 0x1c, 0x91, 0xb9, 0x2c, 0x38, 0xcb,   
 0x9b, 0xd0,   
};

Результат вывода значений полей data на экран:

54 -90 -6 -104 71 -74 -94 5 124 -33  
15655174001060364643  
2434590346  
3409456313  
-101  
-48

**Пример 2**

Массив байт, содержащий значения структуры data:

unsigned char test\_data2[] = {  
 0xc1, 0x3c, 0xd7, 0x6f, 0x85, 0xc8, 0xe3, 0xa9,   
 0x05, 0x05, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   
 0xe3, 0x0d, 0x5e, 0x61, 0xab, 0x8c, 0xb4, 0xcb,   
 0x5b, 0xe6, 0x14, 0x5b, 0x1a, 0xd3, 0xdf, 0xc4,   
 0x3f, 0x72,   
};

Результат вывода значений полей data на экран:

-63 60 -41 111 -123 -56 -29 -87 5 5  
14678511753208139235  
1528096347  
3303002906  
63  
114

## Вариант №5

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| char | 1 | Нет | 1 |
| unsigned short | 8 | Нет | 4 |
| long | 8 | Да | 4 |

Порядок байт от старшего к младшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 char field1;  
 unsigned short field2;  
 char field3;  
 long field4[5];  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для преобразования структуры data в массив байт. В реализации необходимо учесть особенности платформы. Вывести на экран значения массива байт, как показано в примерах ниже:

**Пример 1**

Функция для заполнения полей структуры data:

void test\_code1(struct data \*d) {  
 d->field1 = 163;  
 d->field2 = 6997117904417955386;  
 d->field3 = 207;  
 d->field4[0] = -4208287369703064657;  
 d->field4[1] = 8723319949286462693;  
 d->field4[2] = -2640010639825263300;  
 d->field4[3] = -7915155856400244972;  
 d->field4[4] = 7295950769129140711;  
}

Результат вывода массива байт на экран:

A3 00 00 00 61 1A C1 A9   
68 5B 9E 3A CF 00 00 00   
C5 99 28 EF 08 15 FB AF   
79 0F 75 46 97 4C 54 E5   
DB 5C CC 4F 79 F6 A5 3C   
92 27 B7 AF 55 69 1F 14   
65 40 6C 91 A4 AE 31 E7

**Пример 2**

Функция для заполнения полей структуры data:

void test\_code2(struct data \*d) {  
 d->field1 = 41;  
 d->field2 = 14416437636282503248;  
 d->field3 = 172;  
 d->field4[0] = -6886344686537496928;  
 d->field4[1] = 552181918757682491;  
 d->field4[2] = 3213413508130941435;  
 d->field4[3] = -8734936712853845763;  
 d->field4[4] = 2870765502653953792;  
}

Результат вывода массива байт на экран:

29 00 00 00 C8 11 79 A6   
0D 30 D0 50 AC 00 00 00   
A0 6E C9 FE 36 08 A6 A0   
07 A9 BE 87 8E F1 FD 3B   
2C 98 56 8F 6C 96 29 FB   
86 C7 45 56 DE D4 08 FD   
27 D7 02 03 C3 22 77 00

## Вариант №6

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| char | 1 | Нет | 1 |
| unsigned int | 2 | Нет | 8 |
| double | 8 | Да | 2 |
| unsigned long | 4 | Нет | 1 |

Порядок байт от старшего к младшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 char field1;  
 unsigned int field2;  
 double field3;  
 unsigned long field4;  
 unsigned long field5[10];  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для преобразования структуры data в массив байт. В реализации необходимо учесть особенности платформы. Вывести на экран значения массива байт, как показано в примерах ниже:

**Пример 1**

Функция для заполнения полей структуры data:

void test\_code1(struct data \*d) {  
 d->field1 = 129;  
 d->field2 = 20498;  
 d->field3 = -0.8017298709958558;  
 d->field4 = 2724056614;  
 d->field5[0] = 3592129991;  
 d->field5[1] = 69088631;  
 d->field5[2] = 819417626;  
 d->field5[3] = 3564396062;  
 d->field5[4] = 639510475;  
 d->field5[5] = 1307757782;  
 d->field5[6] = 2787862629;  
 d->field5[7] = 1788189832;  
 d->field5[8] = 2816436257;  
 d->field5[9] = 2898381296;  
}

Результат вывода массива байт на экран:

81 00 00 00 00 00 00 00   
50 12 BF E9 A7 C5 67 04   
E9 78 A2 5D CE 26 D6 1B   
8D C7 04 1E 35 77 30 D7   
52 1A D4 74 5E 1E 26 1E   
27 CB 4D F2 CC D6 A6 2B   
68 65 6A 95 9C 88 A7 DF   
68 21 AC C1 C9 F0

**Пример 2**

Функция для заполнения полей структуры data:

void test\_code2(struct data \*d) {  
 d->field1 = 94;  
 d->field2 = 59029;  
 d->field3 = 0.9935631554575286;  
 d->field4 = 1333765572;  
 d->field5[0] = 1290884761;  
 d->field5[1] = 902573914;  
 d->field5[2] = 352056692;  
 d->field5[3] = 3125278868;  
 d->field5[4] = 3412637923;  
 d->field5[5] = 1423630225;  
 d->field5[6] = 537805279;  
 d->field5[7] = 4082893214;  
 d->field5[8] = 615440088;  
 d->field5[9] = 2106418893;  
}

Результат вывода массива байт на экран:

5E 00 00 00 00 00 00 00   
E6 95 3F EF CB 44 F5 66   
6B B8 4F 7F A5 C4 4C F1   
56 99 35 CC 2F 5A 14 FB   
F5 74 BA 47 F8 94 CB 68   
B8 E3 54 DA DF 91 20 0E   
41 DF F3 5C 01 9E 24 AE   
DE D8 7D 8D 66 CD

## Вариант №7

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| char | 1 | Нет | 1 |
| unsigned short | 2 | Нет | 8 |

Порядок байт от младшего к старшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 char field1[6];  
 char field2;  
 unsigned short field3;  
 char field4;  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для разбора массива байт, содержащего структуру data. В реализации необходимо учесть особенности платформы. Вывести на экран значения полей, как показано в примерах ниже.

**Пример 1**

Массив байт, содержащий значения структуры data:

unsigned char test\_data1[] = {  
 0xf6, 0x51, 0xd0, 0x58, 0xc9, 0x6d, 0x75, 0x00,   
 0x0d, 0xb6, 0x03,   
};

Результат вывода значений полей data на экран:

246 81 208 88 201 109  
117  
46605  
3

**Пример 2**

Массив байт, содержащий значения структуры data:

unsigned char test\_data2[] = {  
 0xee, 0x79, 0x31, 0x8a, 0xfa, 0xbc, 0x13, 0x00,   
 0x59, 0xeb, 0x1e,   
};

Результат вывода значений полей data на экран:

238 121 49 138 250 188  
19  
60249  
30

## Вариант №8

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| unsigned int | 2 | Нет | 4 |
| int | 2 | Да | 4 |
| char | 1 | Да | 1 |

Порядок байт от младшего к старшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 unsigned int field1;  
 int field2[10];  
 char field3;  
 unsigned int field4;  
 int field5;  
 char field6;  
 int field7;  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для разбора массива байт, содержащего структуру data. В реализации необходимо учесть особенности платформы. Вывести на экран значения полей, как показано в примерах ниже.

**Пример 1**

Массив байт, содержащий значения структуры data:

unsigned char test\_data1[] = {  
 0x0a, 0xdb, 0x00, 0x00, 0xa8, 0xef, 0x00, 0x00,   
 0x4a, 0x31, 0x00, 0x00, 0x56, 0xf2, 0x00, 0x00,   
 0x0d, 0xbc, 0x00, 0x00, 0x38, 0x2e, 0x00, 0x00,   
 0x29, 0x86, 0x00, 0x00, 0xad, 0x84, 0x00, 0x00,   
 0x67, 0x00, 0x00, 0x00, 0xf0, 0x01, 0x00, 0x00,   
 0xc9, 0x49, 0xe0, 0x00, 0x1f, 0x8b, 0x00, 0x00,   
 0x28, 0xcc, 0xaf, 0x00, 0x39, 0xc1,   
};

Результат вывода значений полей data на экран:

56074  
-4184 12618 -3498 -17395 11832 -31191 -31571 103 496 18889  
-32  
35615  
-13272  
-81  
-16071

**Пример 2**

Массив байт, содержащий значения структуры data:

unsigned char test\_data2[] = {  
 0xd3, 0x73, 0x00, 0x00, 0x6b, 0x5b, 0x00, 0x00,   
 0x1c, 0x27, 0x00, 0x00, 0xc1, 0x8b, 0x00, 0x00,   
 0x40, 0x31, 0x00, 0x00, 0xd5, 0xdf, 0x00, 0x00,   
 0xba, 0x75, 0x00, 0x00, 0xca, 0xd9, 0x00, 0x00,   
 0x99, 0xea, 0x00, 0x00, 0xaf, 0x42, 0x00, 0x00,   
 0x94, 0x4d, 0x28, 0x00, 0xf8, 0xbb, 0x00, 0x00,   
 0x4d, 0x5f, 0x8c, 0x00, 0x3a, 0x82,   
};

Результат вывода значений полей data на экран:

29651  
23403 10012 -29759 12608 -8235 30138 -9782 -5479 17071 19860  
40  
48120  
24397  
-116  
-32198

## Вариант №9

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| char | 1 | Да | 1 |
| float | 4 | Да | 1 |
| unsigned int | 4 | Нет | 2 |
| long | 4 | Да | 8 |

Порядок байт от младшего к старшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 char field1;  
 float field2[2];  
 char field3;  
 unsigned int field4;  
 long field5;  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для разбора массива байт, содержащего структуру data. В реализации необходимо учесть особенности платформы. Вывести на экран значения полей, как показано в примерах ниже.

**Пример 1**

Массив байт, содержащий значения структуры data:

unsigned char test\_data1[] = {  
 0xd7, 0xc9, 0x31, 0xd5, 0xbd, 0x13, 0x23, 0x2f,   
 0xbe, 0x73, 0x59, 0x23, 0x08, 0x03, 0x00, 0x00,   
 0x9b, 0x29, 0x34, 0x99,   
};

Результат вывода значений полей data на экран:

-41  
-0.1040988638997078 -0.17103223502635956  
115  
50864985  
-1724634725

**Пример 2**

Массив байт, содержащий значения структуры data:

unsigned char test\_data2[] = {  
 0xad, 0x31, 0xc4, 0x65, 0xbf, 0x0c, 0xa6, 0x2f,   
 0xbe, 0x2f, 0xb5, 0xc9, 0xb8, 0x41, 0x00, 0x00,   
 0xa1, 0xaf, 0x81, 0x00,   
};

Результат вывода значений полей data на экран:

-83  
-0.897524893283844 -0.17153185606002808  
47  
1102629301  
8499105

## Вариант №10

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| unsigned short | 2 | Нет | 4 |
| unsigned int | 4 | Нет | 1 |
| char | 1 | Нет | 1 |

Порядок байт от старшего к младшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 unsigned short field1;  
 unsigned short field2[6];  
 unsigned int field3;  
 unsigned int field4;  
 char field5;  
 unsigned int field6;  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для разбора массива байт, содержащего структуру data. В реализации необходимо учесть особенности платформы. Вывести на экран значения полей, как показано в примерах ниже.

**Пример 1**

Массив байт, содержащий значения структуры data:

unsigned char test\_data1[] = {  
 0x4b, 0x32, 0x00, 0x00, 0x7a, 0x99, 0x00, 0x00,   
 0x71, 0x02, 0x00, 0x00, 0x77, 0xa4, 0x00, 0x00,   
 0xde, 0x86, 0x00, 0x00, 0x08, 0x15, 0x00, 0x00,   
 0x4f, 0x4a, 0x08, 0x70, 0x7f, 0xdb, 0x00, 0x39,   
 0x90, 0x13, 0x95, 0x72, 0xb2, 0x4b, 0x5a,   
};

Результат вывода значений полей data на экран:

19250  
31385 28930 30628 56966 2069 20298  
141590491  
3772435  
149  
1924287322

**Пример 2**

Массив байт, содержащий значения структуры data:

unsigned char test\_data2[] = {  
 0x63, 0x9b, 0x00, 0x00, 0xa5, 0xe4, 0x00, 0x00,   
 0x49, 0x7b, 0x00, 0x00, 0xb2, 0x60, 0x00, 0x00,   
 0x35, 0x81, 0x00, 0x00, 0x9b, 0xcd, 0x00, 0x00,   
 0xee, 0xe5, 0x55, 0x29, 0xda, 0x91, 0x45, 0x65,   
 0xa1, 0x0b, 0xce, 0x3d, 0xff, 0xd7, 0x3b,   
};

Результат вывода значений полей data на экран:

25499  
42468 18811 45664 13697 39885 61157  
1428806289  
1164288267  
206  
1040176955

## Вариант №11

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| float | 4 | Да | 2 |
| unsigned int | 4 | Нет | 8 |
| char | 1 | Нет | 1 |

Порядок байт от старшего к младшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 float field1;  
 unsigned int field2;  
 unsigned int field3;  
 char field4[6];  
 char field5;  
 char field6;  
 unsigned int field7[2];  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для разбора массива байт, содержащего структуру data. В реализации необходимо учесть особенности платформы. Вывести на экран значения полей, как показано в примерах ниже.

**Пример 1**

Массив байт, содержащий значения структуры data:

unsigned char test\_data1[] = {  
 0xbe, 0xa3, 0xb5, 0xd0, 0x00, 0x00, 0x00, 0x00,   
 0x65, 0x3d, 0xf0, 0x3e, 0x00, 0x00, 0x00, 0x00,   
 0xb1, 0xe1, 0xf0, 0x36, 0xc9, 0xd0, 0xcf, 0x5b,   
 0x2d, 0x31, 0x93, 0x21, 0x00, 0x00, 0x00, 0x00,   
 0xd4, 0x4b, 0x78, 0xfc, 0x00, 0x00, 0x00, 0x00,   
 0x4e, 0xcd, 0xac, 0xa5,   
};

Результат вывода значений полей data на экран:

-0.3197464942932129  
1698558014  
2984374326  
201 208 207 91 45 49  
147  
33  
3561715964 1322101925

**Пример 2**

Массив байт, содержащий значения структуры data:

unsigned char test\_data2[] = {  
 0x3f, 0x17, 0xec, 0x40, 0x00, 0x00, 0x00, 0x00,   
 0xb2, 0xf1, 0xdb, 0x3c, 0x00, 0x00, 0x00, 0x00,   
 0x69, 0x99, 0x47, 0x54, 0x6d, 0x10, 0x0a, 0x04,   
 0x65, 0x2c, 0x5e, 0x92, 0x00, 0x00, 0x00, 0x00,   
 0xc9, 0xac, 0xf6, 0xf2, 0x00, 0x00, 0x00, 0x00,   
 0x6a, 0x11, 0x3c, 0x52,   
};

Результат вывода значений полей data на экран:

0.5934486389160156  
3002194748  
1771652948  
109 16 10 4 101 44  
94  
146  
3383555826 1779514450

## Вариант №12

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| short | 2 | Да | 4 |
| char | 1 | Да | 1 |
| long | 4 | Да | 4 |

Порядок байт от младшего к старшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 short field1[3];  
 char field2;  
 long field3;  
 long field4;  
 long field5;  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для преобразования структуры data в массив байт. В реализации необходимо учесть особенности платформы. Вывести на экран значения массива байт, как показано в примерах ниже:

**Пример 1**

Функция для заполнения полей структуры data:

void test\_code1(struct data \*d) {  
 d->field1[0] = 13390;  
 d->field1[1] = -15040;  
 d->field1[2] = -29919;  
 d->field2 = -110;  
 d->field3 = 135396032;  
 d->field4 = -947429744;  
 d->field5 = -1156503834;  
}

Результат вывода массива байт на экран:

4E 34 00 00 40 C5 00 00   
21 8B 92 00 C0 FA 11 08   
90 5E 87 C7 E6 26 11 BB

**Пример 2**

Функция для заполнения полей структуры data:

void test\_code2(struct data \*d) {  
 d->field1[0] = 29603;  
 d->field1[1] = 3878;  
 d->field1[2] = 23150;  
 d->field2 = -52;  
 d->field3 = 1430858692;  
 d->field4 = 1167407972;  
 d->field5 = -691989992;  
}

Результат вывода массива байт на экран:

A3 73 00 00 26 0F 00 00   
6E 5A CC 00 C4 2B 49 55   
64 3B 95 45 18 12 C1 D6

## Вариант №13

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| double | 8 | Да | 4 |
| char | 1 | Нет | 1 |
| unsigned short | 2 | Нет | 8 |
| unsigned int | 2 | Нет | 2 |

Порядок байт от младшего к старшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 double field1;  
 char field2[9];  
 unsigned short field3;  
 unsigned short field4;  
 unsigned int field5[7];  
 unsigned int field6;  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для разбора массива байт, содержащего структуру data. В реализации необходимо учесть особенности платформы. Вывести на экран значения полей, как показано в примерах ниже.

**Пример 1**

Массив байт, содержащий значения структуры data:

unsigned char test\_data1[] = {  
 0xf0, 0xe1, 0x7f, 0x56, 0x78, 0x85, 0xd7, 0x3f,   
 0xef, 0x14, 0xc5, 0xc4, 0xa6, 0x3c, 0xdc, 0xcf,   
 0x15, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   
 0x3e, 0xc4, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   
 0x06, 0x2e, 0x1b, 0x65, 0x64, 0x75, 0x84, 0x73,   
 0xf0, 0x17, 0x99, 0xea, 0xc8, 0x9a, 0x81, 0x9c,   
 0x10, 0xac,   
};

Результат вывода значений полей data на экран:

0.3675213665697177  
239 20 197 196 166 60 220 207 21  
50238  
11782  
25883 30052 29572 6128 60057 39624 40065  
44048

**Пример 2**

Массив байт, содержащий значения структуры data:

unsigned char test\_data2[] = {  
 0x18, 0x63, 0x80, 0x5c, 0x89, 0x5a, 0xca, 0xbf,   
 0x46, 0xe0, 0x1b, 0xc1, 0x46, 0xb3, 0x03, 0x58,   
 0x9e, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   
 0x5d, 0xed, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   
 0x1f, 0x3b, 0x9a, 0x4f, 0x4c, 0x75, 0xdc, 0xeb,   
 0x4d, 0x0b, 0xd7, 0xdf, 0x45, 0xee, 0x91, 0x9f,   
 0x43, 0x80,   
};

Результат вывода значений полей data на экран:

-0.20588795677829164  
70 224 27 193 70 179 3 88 158  
60765  
15135  
20378 30028 60380 2893 57303 60997 40849  
32835

## Вариант №14

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| short | 2 | Да | 2 |
| char | 1 | Нет | 1 |
| unsigned int | 4 | Нет | 2 |
| unsigned long long | 8 | Нет | 2 |

Порядок байт от младшего к старшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 short field1[10];  
 char field2[5];  
 unsigned int field3;  
 char field4;  
 unsigned long long field5;  
 char field6;  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для разбора массива байт, содержащего структуру data. В реализации необходимо учесть особенности платформы. Вывести на экран значения полей, как показано в примерах ниже.

**Пример 1**

Массив байт, содержащий значения структуры data:

unsigned char test\_data1[] = {  
 0x5c, 0xba, 0x74, 0xe3, 0x95, 0x72, 0x0a, 0xef,   
 0xbe, 0x7c, 0x69, 0xcd, 0xdf, 0x72, 0x19, 0x17,   
 0x47, 0xbf, 0xc8, 0xae, 0xf4, 0xa3, 0xd7, 0xbb,   
 0xc8, 0x00, 0x24, 0xbb, 0x5d, 0xa8, 0x8e, 0x00,   
 0x55, 0x4d, 0x99, 0x1c, 0xa7, 0xb0, 0xc9, 0x5d,   
 0x4c,   
};

Результат вывода значений полей data на экран:

-17828 -7308 29333 -4342 31934 -12951 29407 5913 -16569 -20792  
244 163 215 187 200  
2824715044  
142  
6758126947631975765  
76

**Пример 2**

Массив байт, содержащий значения структуры data:

unsigned char test\_data2[] = {  
 0x53, 0x37, 0x5b, 0x6a, 0x5a, 0xfa, 0x2d, 0xa5,   
 0xb7, 0x8a, 0x64, 0x67, 0xc3, 0x2e, 0xe6, 0xd0,   
 0xb3, 0x50, 0xfa, 0x73, 0x57, 0xec, 0x34, 0xf0,   
 0x4d, 0x00, 0xb8, 0x44, 0xe6, 0x7a, 0xbe, 0x00,   
 0x83, 0xe1, 0x59, 0x55, 0xf9, 0xda, 0x03, 0xa4,   
 0x87,   
};

Результат вывода значений полей data на экран:

14163 27227 -1446 -23251 -30025 26468 11971 -12058 20659 29690  
87 236 52 240 77  
2061911224  
190  
11818530611563979139  
135

## Вариант №15

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| double | 8 | Да | 4 |
| float | 4 | Да | 1 |
| char | 1 | Нет | 1 |

Порядок байт от старшего к младшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 double field1;  
 float field2;  
 char field3;  
 double field4[3];  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для разбора массива байт, содержащего структуру data. В реализации необходимо учесть особенности платформы. Вывести на экран значения полей, как показано в примерах ниже.

**Пример 1**

Массив байт, содержащий значения структуры data:

unsigned char test\_data1[] = {  
 0x3f, 0xd2, 0xa5, 0x69, 0x0c, 0xf9, 0x88, 0x44,   
 0xbf, 0x4d, 0xec, 0x31, 0x80, 0x00, 0x00, 0x00,   
 0xbf, 0xed, 0xcc, 0x92, 0x44, 0x2f, 0x24, 0xce,   
 0x3f, 0xe1, 0x93, 0x9f, 0xb8, 0x25, 0x69, 0x54,   
 0x3f, 0xea, 0x89, 0x70, 0x78, 0x75, 0xa8, 0x4c,   
};

Результат вывода значений полей data на экран:

0.29134584681571973  
-0.804385244846344  
128  
-0.9312220889779839 0.5492704960445658 0.829277262947707

**Пример 2**

Массив байт, содержащий значения структуры data:

unsigned char test\_data2[] = {  
 0xbf, 0xbe, 0xea, 0x4d, 0x69, 0x81, 0x23, 0xd0,   
 0xbc, 0x8d, 0x0b, 0xa4, 0x7b, 0x00, 0x00, 0x00,   
 0x3f, 0xeb, 0x4a, 0xc5, 0x07, 0x49, 0xf7, 0x10,   
 0xbf, 0xe8, 0x4b, 0xe3, 0x16, 0x15, 0x1b, 0x02,   
 0xbf, 0x7e, 0xa7, 0x5c, 0xc1, 0x43, 0x8d, 0x00,   
};

Результат вывода значений полей data на экран:

-0.12076267076294234  
-0.017217464745044708  
123  
0.8528771536218489 -0.7592635566041681 -0.007483827899906759

## Вариант №16

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| char | 1 | Нет | 1 |
| double | 8 | Да | 1 |
| unsigned short | 4 | Нет | 2 |

Порядок байт от младшего к старшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 char field1[2];  
 double field2;  
 char field3;  
 double field4;  
 char field5;  
 char field6;  
 unsigned short field7;  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для разбора массива байт, содержащего структуру data. В реализации необходимо учесть особенности платформы. Вывести на экран значения полей, как показано в примерах ниже.

**Пример 1**

Массив байт, содержащий значения структуры data:

unsigned char test\_data1[] = {  
 0xc6, 0x4c, 0xec, 0x27, 0xde, 0x8b, 0x09, 0x7c,   
 0xe8, 0xbf, 0xbf, 0xe0, 0x7b, 0xd5, 0x7d, 0x6d,   
 0x7e, 0xcb, 0xbf, 0x1b, 0xdd, 0x00, 0x59, 0x92,   
 0x84, 0x1a,   
};

Результат вывода значений полей data на экран:

198 76  
-0.7651412708084968  
191  
-0.21479576725229688  
27  
221  
444895833

**Пример 2**

Массив байт, содержащий значения структуры data:

unsigned char test\_data2[] = {  
 0x04, 0x30, 0x1c, 0x14, 0x95, 0xf7, 0x94, 0xac,   
 0xeb, 0x3f, 0x24, 0xc0, 0x26, 0xa7, 0xe7, 0x76,   
 0x63, 0xc2, 0x3f, 0x93, 0x04, 0x00, 0x56, 0x26,   
 0xda, 0x78,   
};

Результат вывода значений полей data на экран:

4 48  
0.8648171268074551  
36  
0.14366041480234593  
147  
4  
2027562582

## Вариант №17

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| char | 1 | Да | 1 |
| float | 4 | Да | 4 |
| unsigned short | 2 | Нет | 1 |

Порядок байт от младшего к старшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 char field1[6];  
 char field2;  
 float field3;  
 char field4;  
 unsigned short field5[4];  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для разбора массива байт, содержащего структуру data. В реализации необходимо учесть особенности платформы. Вывести на экран значения полей, как показано в примерах ниже.

**Пример 1**

Массив байт, содержащий значения структуры data:

unsigned char test\_data1[] = {  
 0xe9, 0xf9, 0x74, 0x93, 0x91, 0x44, 0x1d, 0x00,   
 0xac, 0x7d, 0x4e, 0x3f, 0x5a, 0xe0, 0xea, 0x0f,   
 0xd3, 0xe8, 0xa1, 0x23, 0xfb,   
};

Результат вывода значений полей data на экран:

-23 -7 116 -109 -111 68  
29  
0.8066051006317139  
90  
60128 54031 41448 64291

**Пример 2**

Массив байт, содержащий значения структуры data:

unsigned char test\_data2[] = {  
 0xf1, 0xee, 0xc7, 0x01, 0x81, 0x32, 0x29, 0x00,   
 0xac, 0xe0, 0x86, 0xbe, 0x4a, 0xa6, 0x00, 0xa1,   
 0xe7, 0x00, 0xf8, 0x82, 0x84,   
};

Результат вывода значений полей data на экран:

-15 -18 -57 1 -127 50  
41  
-0.2634328603744507  
74  
166 59297 63488 33922

## Вариант №18

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| char | 1 | Нет | 1 |
| unsigned long long | 8 | Нет | 2 |
| int | 2 | Да | 4 |

Порядок байт от младшего к старшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 char field1;  
 unsigned long long field2[4];  
 int field3;  
 char field4;  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для преобразования структуры data в массив байт. В реализации необходимо учесть особенности платформы. Вывести на экран значения массива байт, как показано в примерах ниже:

**Пример 1**

Функция для заполнения полей структуры data:

void test\_code1(struct data \*d) {  
 d->field1 = 204;  
 d->field2[0] = 8429912020236203225;  
 d->field2[1] = 17410251274451664244;  
 d->field2[2] = 6239122298411886413;  
 d->field2[3] = 3278070707572838231;  
 d->field3 = 25668;  
 d->field4 = 211;  
}

Результат вывода массива байт на экран:

CC 00 D9 74 F9 9D 51 10   
FD 74 74 BD FB 67 4A A3   
9D F1 4D 27 0F 93 AC D0   
95 56 57 0F 7E C9 F0 0B   
7E 2D 00 00 44 64 D3

**Пример 2**

Функция для заполнения полей структуры data:

void test\_code2(struct data \*d) {  
 d->field1 = 135;  
 d->field2[0] = 5254238628836538756;  
 d->field2[1] = 8882608672744602989;  
 d->field2[2] = 2675143859622697685;  
 d->field2[3] = 17165320354298522829;  
 d->field3 = 10719;  
 d->field4 = 73;  
}

Результат вывода массива байт на экран:

87 00 84 F9 EF DC 31 CE   
EA 48 6D FD BE 99 86 5D   
45 7B D5 DA E0 B3 2A 05   
20 25 CD EC BE 9D EA 77   
37 EE 00 00 DF 29 49

## Вариант №19

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| long long | 8 | Да | 4 |
| char | 1 | Нет | 1 |
| float | 4 | Да | 4 |

Порядок байт от младшего к старшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 long long field1;  
 char field2;  
 float field3[4];  
 long long field4;  
 float field5;  
 char field6;  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для разбора массива байт, содержащего структуру data. В реализации необходимо учесть особенности платформы. Вывести на экран значения полей, как показано в примерах ниже.

**Пример 1**

Массив байт, содержащий значения структуры data:

unsigned char test\_data1[] = {  
 0x75, 0x94, 0xdd, 0x59, 0xca, 0xee, 0x05, 0x2f,   
 0x9e, 0x00, 0x00, 0x00, 0xba, 0x51, 0x80, 0xbe,   
 0xfd, 0x59, 0xdb, 0x3e, 0xf4, 0xa3, 0x1b, 0xbf,   
 0x65, 0xbe, 0x42, 0x3d, 0x69, 0xa9, 0xb2, 0x31,   
 0x0e, 0x82, 0x82, 0x6b, 0x90, 0x5e, 0xa7, 0xbe,   
 0x26,   
};

Результат вывода значений полей data на экран:

3388376847524664437  
158  
-0.25062352418899536 0.42842093110084534 -0.6079704761505127 0.04754485562443733  
7746897306505619817  
-0.32689332962036133  
38

**Пример 2**

Массив байт, содержащий значения структуры data:

unsigned char test\_data2[] = {  
 0xa2, 0x85, 0xca, 0x97, 0x29, 0x40, 0x51, 0xd6,   
 0x1d, 0x00, 0x00, 0x00, 0xcf, 0xdd, 0xc7, 0x3d,   
 0xda, 0x5e, 0x18, 0x3f, 0x70, 0x45, 0x1b, 0xbf,   
 0x52, 0xb9, 0x1e, 0xbf, 0xc8, 0x51, 0x2f, 0xa7,   
 0x53, 0xce, 0x8f, 0x4e, 0x98, 0x8e, 0xda, 0xbd,   
 0xbf,   
};

Результат вывода значений полей data на экран:

-3003548929094941278  
29  
0.09759103506803513 0.595197319984436 -0.6065282821655273 -0.6200152635574341  
5660970115310506440  
-0.10671728849411011  
191

## Вариант №20

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| char | 1 | Да | 1 |
| short | 4 | Да | 4 |
| unsigned long long | 8 | Нет | 4 |
| float | 4 | Да | 2 |

Порядок байт от старшего к младшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 char field1[7];  
 char field2;  
 char field3;  
 short field4;  
 unsigned long long field5;  
 short field6;  
 float field7;  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для преобразования структуры data в массив байт. В реализации необходимо учесть особенности платформы. Вывести на экран значения массива байт, как показано в примерах ниже:

**Пример 1**

Функция для заполнения полей структуры data:

void test\_code1(struct data \*d) {  
 d->field1[0] = 63;  
 d->field1[1] = 98;  
 d->field1[2] = 64;  
 d->field1[3] = 74;  
 d->field1[4] = 1;  
 d->field1[5] = 39;  
 d->field1[6] = 48;  
 d->field2 = -69;  
 d->field3 = 126;  
 d->field4 = 1325701713;  
 d->field5 = 1944286432146007881;  
 d->field6 = 1243623167;  
 d->field7 = 0.6119346618652344;  
}

Результат вывода массива байт на экран:

3F 62 40 4A 01 27 30 BB   
7E 00 00 00 4F 04 9A 51   
1A FB 7E 35 71 AC A7 49   
4A 20 2E FF 3F 1C A7 C0

**Пример 2**

Функция для заполнения полей структуры data:

void test\_code2(struct data \*d) {  
 d->field1[0] = -50;  
 d->field1[1] = -89;  
 d->field1[2] = -119;  
 d->field1[3] = -26;  
 d->field1[4] = 66;  
 d->field1[5] = -50;  
 d->field1[6] = -11;  
 d->field2 = 16;  
 d->field3 = -94;  
 d->field4 = -959854631;  
 d->field5 = 6904295750977084360;  
 d->field6 = -478288546;  
 d->field7 = -0.11031660437583923;  
}

Результат вывода массива байт на экран:

CE A7 89 E6 42 CE F5 10   
A2 00 00 00 C6 C9 C7 D9   
5F D0 FC 67 AD 3A 5B C8   
E3 7D E5 5E BD E1 ED AC

## Вариант №21

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| unsigned int | 2 | Нет | 4 |
| char | 1 | Нет | 1 |
| int | 2 | Да | 4 |
| long long | 8 | Да | 8 |

Порядок байт от старшего к младшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 unsigned int field1;  
 char field2;  
 int field3[5];  
 unsigned int field4;  
 long long field5;  
 int field6[4];  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для разбора массива байт, содержащего структуру data. В реализации необходимо учесть особенности платформы. Вывести на экран значения полей, как показано в примерах ниже.

**Пример 1**

Массив байт, содержащий значения структуры data:

unsigned char test\_data1[] = {  
 0xbd, 0x08, 0x6d, 0x00, 0xf9, 0x0d, 0x00, 0x00,   
 0x7f, 0x25, 0x00, 0x00, 0xd9, 0x54, 0x00, 0x00,   
 0x74, 0xea, 0x00, 0x00, 0x05, 0x66, 0x00, 0x00,   
 0xb7, 0x3c, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   
 0x6e, 0xdf, 0x8a, 0xfe, 0x49, 0xb9, 0xd6, 0x8c,   
 0xa5, 0xba, 0x00, 0x00, 0x47, 0x9f, 0x00, 0x00,   
 0x37, 0xd7, 0x00, 0x00, 0x2f, 0xc1,   
};

Результат вывода значений полей data на экран:

48392  
109  
-1779 32549 -9900 29930 1382  
46908  
7989257088741791372  
-23110 18335 14295 12225

**Пример 2**

Массив байт, содержащий значения структуры data:

unsigned char test\_data2[] = {  
 0x05, 0x0d, 0xe3, 0x00, 0xd5, 0x9a, 0x00, 0x00,   
 0xbc, 0xd4, 0x00, 0x00, 0x6e, 0xf0, 0x00, 0x00,   
 0xf0, 0xbd, 0x00, 0x00, 0x2d, 0x02, 0x00, 0x00,   
 0x6b, 0xe5, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   
 0x9f, 0xd6, 0x8f, 0xe7, 0xf9, 0xa1, 0xe6, 0x14,   
 0x0e, 0x46, 0x00, 0x00, 0x2f, 0x00, 0x00, 0x00,   
 0x16, 0xa5, 0x00, 0x00, 0xf7, 0x45,   
};

Результат вывода значений полей data на экран:

1293  
227  
-10854 -17196 28400 -3907 11522  
27621  
-6929192750174575084  
3654 12032 5797 -2235

## Вариант №22

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| unsigned long | 8 | Нет | 8 |
| char | 1 | Нет | 1 |
| short | 2 | Да | 4 |

Порядок байт от старшего к младшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 unsigned long field1;  
 char field2;  
 unsigned long field3;  
 char field4;  
 char field5[10];  
 short field6[3];  
 short field7;  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для преобразования структуры data в массив байт. В реализации необходимо учесть особенности платформы. Вывести на экран значения массива байт, как показано в примерах ниже:

**Пример 1**

Функция для заполнения полей структуры data:

void test\_code1(struct data \*d) {  
 d->field1 = 11560181724849911632;  
 d->field2 = 136;  
 d->field3 = 5399079184648041772;  
 d->field4 = 230;  
 d->field5[0] = 16;  
 d->field5[1] = 77;  
 d->field5[2] = 228;  
 d->field5[3] = 181;  
 d->field5[4] = 236;  
 d->field5[5] = 212;  
 d->field5[6] = 236;  
 d->field5[7] = 56;  
 d->field5[8] = 221;  
 d->field5[9] = 35;  
 d->field6[0] = -29330;  
 d->field6[1] = 25201;  
 d->field6[2] = 6064;  
 d->field7 = -15005;  
}

Результат вывода массива байт на экран:

A0 6E 04 07 C0 A4 37 50   
88 00 00 00 00 00 00 00   
4A ED 61 E9 B3 26 F5 2C   
E6 10 4D E4 B5 EC D4 EC   
38 DD 23 00 8D 6E 00 00   
62 71 00 00 17 B0 00 00   
C5 63

**Пример 2**

Функция для заполнения полей структуры data:

void test\_code2(struct data \*d) {  
 d->field1 = 5583602365518111534;  
 d->field2 = 189;  
 d->field3 = 10570860986217070180;  
 d->field4 = 237;  
 d->field5[0] = 194;  
 d->field5[1] = 26;  
 d->field5[2] = 63;  
 d->field5[3] = 121;  
 d->field5[4] = 120;  
 d->field5[5] = 72;  
 d->field5[6] = 211;  
 d->field5[7] = 180;  
 d->field5[8] = 95;  
 d->field5[9] = 242;  
 d->field6[0] = -27553;  
 d->field6[1] = -6472;  
 d->field6[2] = -22172;  
 d->field7 = 20207;  
}

Результат вывода массива байт на экран:

4D 7C F0 C4 AB BC E3 2E   
BD 00 00 00 00 00 00 00   
92 B3 3E 0F 65 80 FE 64   
ED C2 1A 3F 79 78 48 D3   
B4 5F F2 00 94 5F 00 00   
E6 B8 00 00 A9 64 00 00   
4E EF

## Вариант №23

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| char | 1 | Нет | 1 |
| unsigned short | 4 | Нет | 2 |

Порядок байт от младшего к старшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 char field1;  
 unsigned short field2;  
 unsigned short field3[5];  
 unsigned short field4[9];  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для преобразования структуры data в массив байт. В реализации необходимо учесть особенности платформы. Вывести на экран значения массива байт, как показано в примерах ниже:

**Пример 1**

Функция для заполнения полей структуры data:

void test\_code1(struct data \*d) {  
 d->field1 = 247;  
 d->field2 = 4251062887;  
 d->field3[0] = 1333080194;  
 d->field3[1] = 4166287925;  
 d->field3[2] = 371777756;  
 d->field3[3] = 438470582;  
 d->field3[4] = 3694655124;  
 d->field4[0] = 375061747;  
 d->field4[1] = 892879794;  
 d->field4[2] = 586977276;  
 d->field4[3] = 3056102803;  
 d->field4[4] = 3829102729;  
 d->field4[5] = 792456992;  
 d->field4[6] = 1223219516;  
 d->field4[7] = 1673684611;  
 d->field4[8] = 3253187813;  
}

Результат вывода массива байт на экран:

F7 00 67 12 62 FD 82 30   
75 4F 35 82 54 F8 DC E0   
28 16 B6 87 22 1A 94 F6   
37 DC F3 FC 5A 16 B2 43   
38 35 FC 8F FC 22 93 6D   
28 B6 89 78 3B E4 20 EF   
3B 2F 3C D9 E8 48 83 66   
C2 63 E5 B4 E7 C1

**Пример 2**

Функция для заполнения полей структуры data:

void test\_code2(struct data \*d) {  
 d->field1 = 88;  
 d->field2 = 1078253695;  
 d->field3[0] = 3531963062;  
 d->field3[1] = 683819679;  
 d->field3[2] = 138872569;  
 d->field3[3] = 1806509083;  
 d->field3[4] = 2621885573;  
 d->field4[0] = 4078762146;  
 d->field4[1] = 258463676;  
 d->field4[2] = 1954270757;  
 d->field4[3] = 923715682;  
 d->field4[4] = 1732470635;  
 d->field4[5] = 1213872606;  
 d->field4[6] = 457928152;  
 d->field4[7] = 1408017981;  
 d->field4[8] = 1130483688;  
}

Результат вывода массива байт на экран:

58 00 7F D8 44 40 B6 7A   
85 D2 9F 42 C2 28 F9 06   
47 08 1B 24 AD 6B 85 CC   
46 9C A2 F8 1C F3 BC D7   
67 0F 25 CE 7B 74 62 C8   
0E 37 6B 67 43 67 DE 39   
5A 48 D8 6D 4B 1B 3D A6   
EC 53 E8 CF 61 43

## Вариант №24

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| char | 1 | Да | 1 |
| double | 8 | Да | 4 |

Порядок байт от старшего к младшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 char field1[3];  
 double field2;  
 char field3;  
 char field4[5];  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для преобразования структуры data в массив байт. В реализации необходимо учесть особенности платформы. Вывести на экран значения массива байт, как показано в примерах ниже:

**Пример 1**

Функция для заполнения полей структуры data:

void test\_code1(struct data \*d) {  
 d->field1[0] = -109;  
 d->field1[1] = 73;  
 d->field1[2] = -49;  
 d->field2 = -0.018028572248764974;  
 d->field3 = -114;  
 d->field4[0] = 35;  
 d->field4[1] = -113;  
 d->field4[2] = 62;  
 d->field4[3] = 35;  
 d->field4[4] = -38;  
}

Результат вывода массива байт на экран:

93 49 CF 00 BF 92 76 15   
00 CE DE 00 8E 23 8F 3E   
23 DA

**Пример 2**

Функция для заполнения полей структуры data:

void test\_code2(struct data \*d) {  
 d->field1[0] = 28;  
 d->field1[1] = -100;  
 d->field1[2] = -126;  
 d->field2 = 0.9582868596696335;  
 d->field3 = 40;  
 d->field4[0] = -48;  
 d->field4[1] = 58;  
 d->field4[2] = 27;  
 d->field4[3] = 78;  
 d->field4[4] = -6;  
}

Результат вывода массива байт на экран:

1C 9C 82 00 3F EE AA 49   
34 4E F6 B8 28 D0 3A 1B   
4E FA

## Вариант №25

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| unsigned int | 2 | Нет | 4 |
| long long | 8 | Да | 4 |
| char | 1 | Да | 1 |
| unsigned long | 4 | Нет | 4 |

Порядок байт от младшего к старшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 unsigned int field1;  
 long long field2;  
 long long field3[2];  
 char field4;  
 unsigned long field5[4];  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для разбора массива байт, содержащего структуру data. В реализации необходимо учесть особенности платформы. Вывести на экран значения полей, как показано в примерах ниже.

**Пример 1**

Массив байт, содержащий значения структуры data:

unsigned char test\_data1[] = {  
 0xe5, 0xe2, 0x00, 0x00, 0x18, 0x5c, 0xd2, 0x84,   
 0xed, 0xab, 0x03, 0x03, 0x8b, 0x01, 0x6a, 0x38,   
 0x28, 0x77, 0xd5, 0x52, 0x0a, 0xee, 0x39, 0xee,   
 0x96, 0x43, 0x06, 0xaa, 0x87, 0x00, 0x00, 0x00,   
 0x76, 0x71, 0x23, 0xc2, 0x4a, 0x12, 0xbb, 0xad,   
 0x60, 0xc6, 0x3a, 0xe4, 0xc5, 0xe8, 0xc9, 0xcb,   
};

Результат вывода значений полей data на экран:

58085  
217206243667893272  
5968807895778328971 -6195189921880609270  
-121  
3257102710 2914718282 3829057120 3419007173

**Пример 2**

Массив байт, содержащий значения структуры data:

unsigned char test\_data2[] = {  
 0xd4, 0xc5, 0x00, 0x00, 0x96, 0xc1, 0x81, 0xd3,   
 0xe1, 0x8b, 0x64, 0xb1, 0xf6, 0x7c, 0x41, 0xa4,   
 0x04, 0x5f, 0x5c, 0x53, 0xd8, 0x6b, 0xc3, 0xe8,   
 0xe2, 0x4f, 0x0a, 0x94, 0xa1, 0x00, 0x00, 0x00,   
 0x76, 0x7b, 0xa1, 0x6e, 0x6a, 0x92, 0x23, 0x4a,   
 0x13, 0x00, 0x5c, 0x0e, 0x86, 0xba, 0xd9, 0x88,   
};

Результат вывода значений полей data на экран:

50644  
-5664248629292842602  
6006780476545662198 -7779317570342786088  
-95  
1856076662 1243845226 240910355 2295970438

## Вариант №26

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| char | 1 | Да | 1 |
| unsigned long long | 8 | Нет | 4 |
| unsigned int | 2 | Нет | 4 |

Порядок байт от младшего к старшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 char field1[2];  
 unsigned long long field2;  
 char field3[4];  
 char field4;  
 unsigned long long field5;  
 unsigned int field6[3];  
 unsigned int field7;  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для преобразования структуры data в массив байт. В реализации необходимо учесть особенности платформы. Вывести на экран значения массива байт, как показано в примерах ниже:

**Пример 1**

Функция для заполнения полей структуры data:

void test\_code1(struct data \*d) {  
 d->field1[0] = 17;  
 d->field1[1] = -27;  
 d->field2 = 12745158503502439979;  
 d->field3[0] = 110;  
 d->field3[1] = -48;  
 d->field3[2] = 54;  
 d->field3[3] = 16;  
 d->field4 = -84;  
 d->field5 = 1494353453141597844;  
 d->field6[0] = 19831;  
 d->field6[1] = 33659;  
 d->field6[2] = 12472;  
 d->field7 = 49147;  
}

Результат вывода массива байт на экран:

11 E5 00 00 2B E6 4F D7   
21 E6 DF B0 6E D0 36 10   
AC 00 00 00 94 1E 62 57   
8C 02 BD 14 77 4D 00 00   
7B 83 00 00 B8 30 00 00   
FB BF

**Пример 2**

Функция для заполнения полей структуры data:

void test\_code2(struct data \*d) {  
 d->field1[0] = -113;  
 d->field1[1] = -73;  
 d->field2 = 7772834040224777775;  
 d->field3[0] = -119;  
 d->field3[1] = 24;  
 d->field3[2] = -17;  
 d->field3[3] = -12;  
 d->field4 = -82;  
 d->field5 = 6055479180523463806;  
 d->field6[0] = 47182;  
 d->field6[1] = 13724;  
 d->field6[2] = 49077;  
 d->field7 = 5297;  
}

Результат вывода массива байт на экран:

8F B7 00 00 2F 86 A9 99   
60 A7 DE 6B 89 18 EF F4   
AE 00 00 00 7E E4 DF 3B   
3B 62 09 54 4E B8 00 00   
9C 35 00 00 B5 BF 00 00   
B1 14

## Вариант №27

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| float | 4 | Да | 1 |
| char | 1 | Нет | 1 |
| unsigned long long | 8 | Нет | 8 |

Порядок байт от старшего к младшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 float field1;  
 char field2;  
 char field3[5];  
 unsigned long long field4;  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для разбора массива байт, содержащего структуру data. В реализации необходимо учесть особенности платформы. Вывести на экран значения полей, как показано в примерах ниже.

**Пример 1**

Массив байт, содержащий значения структуры data:

unsigned char test\_data1[] = {  
 0x3d, 0xef, 0x01, 0x07, 0xde, 0x5b, 0x06, 0x04,   
 0xfe, 0xdf, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   
 0xfa, 0x8f, 0xdf, 0xca, 0x37, 0x58, 0x9d, 0x90,   
};

Результат вывода значений полей data на экран:

0.11670117825269699  
222  
91 6 4 254 223  
18054895490756550032

**Пример 2**

Массив байт, содержащий значения структуры data:

unsigned char test\_data2[] = {  
 0x3f, 0x13, 0xd5, 0x40, 0x9c, 0xc6, 0x25, 0xd4,   
 0x5a, 0xcd, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   
 0x20, 0x0d, 0x74, 0x05, 0x9a, 0x1b, 0x26, 0x29,   
};

Результат вывода значений полей data на экран:

0.5774726867675781  
156  
198 37 212 90 205  
2309629751320061481

## Вариант №28

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| unsigned short | 2 | Нет | 1 |
| unsigned long | 4 | Нет | 8 |
| char | 1 | Да | 1 |

Порядок байт от старшего к младшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 unsigned short field1;  
 unsigned long field2;  
 unsigned short field3[7];  
 char field4;  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для разбора массива байт, содержащего структуру data. В реализации необходимо учесть особенности платформы. Вывести на экран значения полей, как показано в примерах ниже.

**Пример 1**

Массив байт, содержащий значения структуры data:

unsigned char test\_data1[] = {  
 0xbb, 0x92, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   
 0x93, 0xb9, 0xcc, 0xca, 0xe7, 0xd9, 0x42, 0x91,   
 0xc0, 0x27, 0x08, 0x5a, 0x58, 0x63, 0x69, 0x1c,   
 0xd2, 0x1b, 0x01,   
};

Результат вывода значений полей data на экран:

48018  
2478427338  
59353 17041 49191 2138 22627 26908 53787  
1

**Пример 2**

Массив байт, содержащий значения структуры data:

unsigned char test\_data2[] = {  
 0x02, 0x4a, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   
 0x1a, 0xae, 0xbd, 0x5f, 0x0c, 0xdb, 0x70, 0x0a,   
 0xb2, 0x1d, 0x40, 0x54, 0xac, 0x9b, 0xf0, 0x65,   
 0x05, 0x75, 0xea,   
};

Результат вывода значений полей data на экран:

586  
447659359  
3291 28682 45597 16468 44187 61541 1397  
-22

## Вариант №29

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| unsigned short | 2 | Нет | 8 |
| char | 1 | Да | 1 |

Порядок байт от младшего к старшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 unsigned short field1;  
 unsigned short field2[2];  
 char field3[10];  
 char field4[10];  
 unsigned short field5;  
 char field6;  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для преобразования структуры data в массив байт. В реализации необходимо учесть особенности платформы. Вывести на экран значения массива байт, как показано в примерах ниже:

**Пример 1**

Функция для заполнения полей структуры data:

void test\_code1(struct data \*d) {  
 d->field1 = 44085;  
 d->field2[0] = 59209;  
 d->field2[1] = 56659;  
 d->field3[0] = 39;  
 d->field3[1] = -95;  
 d->field3[2] = 82;  
 d->field3[3] = 88;  
 d->field3[4] = -88;  
 d->field3[5] = -25;  
 d->field3[6] = 82;  
 d->field3[7] = 79;  
 d->field3[8] = 117;  
 d->field3[9] = 110;  
 d->field4[0] = -71;  
 d->field4[1] = -13;  
 d->field4[2] = -26;  
 d->field4[3] = 102;  
 d->field4[4] = 61;  
 d->field4[5] = 108;  
 d->field4[6] = -12;  
 d->field4[7] = 22;  
 d->field4[8] = -36;  
 d->field4[9] = -107;  
 d->field5 = 53243;  
 d->field6 = 115;  
}

Результат вывода массива байт на экран:

35 AC 00 00 00 00 00 00   
49 E7 00 00 00 00 00 00   
53 DD 27 A1 52 58 A8 E7   
52 4F 75 6E B9 F3 E6 66   
3D 6C F4 16 DC 95 00 00   
FB CF 73

**Пример 2**

Функция для заполнения полей структуры data:

void test\_code2(struct data \*d) {  
 d->field1 = 59585;  
 d->field2[0] = 10099;  
 d->field2[1] = 38942;  
 d->field3[0] = 77;  
 d->field3[1] = -127;  
 d->field3[2] = -44;  
 d->field3[3] = 75;  
 d->field3[4] = 30;  
 d->field3[5] = -55;  
 d->field3[6] = -115;  
 d->field3[7] = -70;  
 d->field3[8] = -105;  
 d->field3[9] = -127;  
 d->field4[0] = 14;  
 d->field4[1] = 78;  
 d->field4[2] = -52;  
 d->field4[3] = -16;  
 d->field4[4] = -98;  
 d->field4[5] = -113;  
 d->field4[6] = -55;  
 d->field4[7] = 8;  
 d->field4[8] = 56;  
 d->field4[9] = -7;  
 d->field5 = 49481;  
 d->field6 = 74;  
}

Результат вывода массива байт на экран:

C1 E8 00 00 00 00 00 00   
73 27 00 00 00 00 00 00   
1E 98 4D 81 D4 4B 1E C9   
8D BA 97 81 0E 4E CC F0   
9E 8F C9 08 38 F9 00 00   
49 C1 4A

## Вариант №30

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| char | 1 | Да | 1 |
| float | 4 | Да | 2 |
| unsigned long | 8 | Нет | 8 |
| long long | 8 | Да | 8 |

Порядок байт от младшего к старшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 char field1;  
 float field2;  
 char field3;  
 unsigned long field4;  
 long long field5[3];  
 unsigned long field6;  
 long long field7;  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для разбора массива байт, содержащего структуру data. В реализации необходимо учесть особенности платформы. Вывести на экран значения полей, как показано в примерах ниже.

**Пример 1**

Массив байт, содержащий значения структуры data:

unsigned char test\_data1[] = {  
 0x1f, 0x00, 0x9a, 0x10, 0x74, 0x3f, 0xa1, 0x00,   
 0x1e, 0x83, 0x8e, 0xb6, 0x94, 0x3c, 0x1b, 0x46,   
 0xc3, 0x22, 0xc7, 0x69, 0x7b, 0x08, 0xc2, 0x05,   
 0xcf, 0x22, 0xd8, 0x46, 0x20, 0x01, 0x44, 0xa5,   
 0x08, 0xe8, 0x3d, 0x39, 0xaf, 0x7b, 0x6d, 0xce,   
 0x54, 0xa1, 0xde, 0x65, 0x80, 0xdd, 0x30, 0xa7,   
 0xd4, 0x27, 0xe1, 0x3f, 0xb8, 0x8f, 0xc7, 0x95,   
};

Результат вывода значений полей data на экран:

31  
0.9533783197402954  
-95  
5051698016441762590  
414903441820164803 -6538099520895966513 -3572062936925083640  
12047372546750718292  
-7653991020184393772

**Пример 2**

Массив байт, содержащий значения структуры data:

unsigned char test\_data2[] = {  
 0xed, 0x00, 0x73, 0x26, 0xad, 0xbe, 0xec, 0x00,   
 0x0c, 0x04, 0xb7, 0xe5, 0x24, 0xa7, 0x18, 0x35,   
 0x90, 0x5e, 0xdb, 0xec, 0xb2, 0x8d, 0xa1, 0xe4,   
 0x2c, 0x21, 0xfd, 0x1e, 0xd5, 0x1c, 0xac, 0x21,   
 0x00, 0x10, 0xc7, 0xb8, 0x91, 0x48, 0x54, 0xd7,   
 0xdc, 0xcf, 0x25, 0x52, 0x0a, 0xc5, 0xee, 0xb0,   
 0x66, 0xd3, 0xfe, 0xbe, 0x03, 0x9d, 0x10, 0x24,   
};

Результат вывода значений полей data на экран:

-19  
-0.3381839692592621  
-20  
3825991660365874188  
-1972139362194071920 2426346000919372076 -2930637666803838976  
12749344243251007452  
2598749624407610214

## Вариант №31

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| short | 2 | Да | 4 |
| char | 1 | Да | 1 |
| unsigned long | 8 | Нет | 4 |

Порядок байт от младшего к старшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 short field1;  
 char field2[8];  
 char field3;  
 unsigned long field4;  
 short field5[4];  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для преобразования структуры data в массив байт. В реализации необходимо учесть особенности платформы. Вывести на экран значения массива байт, как показано в примерах ниже:

**Пример 1**

Функция для заполнения полей структуры data:

void test\_code1(struct data \*d) {  
 d->field1 = 4812;  
 d->field2[0] = 7;  
 d->field2[1] = -54;  
 d->field2[2] = 68;  
 d->field2[3] = -49;  
 d->field2[4] = 57;  
 d->field2[5] = -120;  
 d->field2[6] = 1;  
 d->field2[7] = 125;  
 d->field3 = -9;  
 d->field4 = 2463757968177348877;  
 d->field5[0] = -3295;  
 d->field5[1] = -8088;  
 d->field5[2] = 2869;  
 d->field5[3] = 20935;  
}

Результат вывода массива байт на экран:

CC 12 07 CA 44 CF 39 88   
01 7D F7 00 0D C1 BF 89   
D1 06 31 22 21 F3 00 00   
68 E0 00 00 35 0B 00 00   
C7 51

**Пример 2**

Функция для заполнения полей структуры data:

void test\_code2(struct data \*d) {  
 d->field1 = -10082;  
 d->field2[0] = -68;  
 d->field2[1] = 47;  
 d->field2[2] = -49;  
 d->field2[3] = 63;  
 d->field2[4] = -115;  
 d->field2[5] = 23;  
 d->field2[6] = 25;  
 d->field2[7] = -102;  
 d->field3 = 53;  
 d->field4 = 16114501553892927875;  
 d->field5[0] = -3181;  
 d->field5[1] = 15790;  
 d->field5[2] = 23438;  
 d->field5[3] = 6851;  
}

Результат вывода массива байт на экран:

9E D8 BC 2F CF 3F 8D 17   
19 9A 35 00 83 A9 44 F4   
C8 35 A2 DF 93 F3 00 00   
AE 3D 00 00 8E 5B 00 00   
C3 1A

## Вариант №32

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| char | 1 | Нет | 1 |
| short | 2 | Да | 4 |

Порядок байт от младшего к старшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 char field1;  
 char field2;  
 short field3;  
 char field4;  
 short field5;  
 char field6;  
 char field7[6];  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для разбора массива байт, содержащего структуру data. В реализации необходимо учесть особенности платформы. Вывести на экран значения полей, как показано в примерах ниже.

**Пример 1**

Массив байт, содержащий значения структуры data:

unsigned char test\_data1[] = {  
 0x5f, 0xe0, 0x00, 0x00, 0x2e, 0x86, 0x50, 0x00,   
 0xb6, 0x8c, 0x43, 0xc8, 0x9e, 0x1b, 0xbb, 0xa0,   
 0x37,   
};

Результат вывода значений полей data на экран:

95  
224  
-31186  
80  
-29514  
67  
200 158 27 187 160 55

**Пример 2**

Массив байт, содержащий значения структуры data:

unsigned char test\_data2[] = {  
 0xcd, 0x5b, 0x00, 0x00, 0x2c, 0x23, 0x68, 0x00,   
 0x23, 0x51, 0x6e, 0xdd, 0x77, 0x9e, 0x64, 0x5c,   
 0xbc,   
};

Результат вывода значений полей data на экран:

205  
91  
9004  
104  
20771  
110  
221 119 158 100 92 188

## Вариант №33

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| char | 1 | Нет | 1 |
| int | 8 | Да | 1 |
| long long | 8 | Да | 2 |
| short | 2 | Да | 1 |

Порядок байт от старшего к младшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 char field1;  
 int field2;  
 long long field3;  
 char field4[10];  
 short field5[9];  
 char field6;  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для разбора массива байт, содержащего структуру data. В реализации необходимо учесть особенности платформы. Вывести на экран значения полей, как показано в примерах ниже.

**Пример 1**

Массив байт, содержащий значения структуры data:

unsigned char test\_data1[] = {  
 0x83, 0xd7, 0xd1, 0x1e, 0x16, 0x8d, 0xd3, 0x0a,   
 0xc3, 0x00, 0xb5, 0xa9, 0x17, 0x8f, 0x04, 0xd9,   
 0x0a, 0x08, 0x81, 0xe1, 0x11, 0x1c, 0xde, 0x4c,   
 0x9c, 0xb1, 0x72, 0x56, 0x79, 0x0f, 0x06, 0x88,   
 0x1c, 0x1f, 0x40, 0x35, 0x1e, 0xf5, 0x77, 0xaf,   
 0xef, 0xec, 0xa4, 0x7e, 0xdd, 0xcf, 0x59,   
};

Результат вывода значений полей data на экран:

131  
-2895500003204986173  
-5356724378751399416  
129 225 17 28 222 76 156 177 114 86  
30991 1672 7199 16437 7925 30639 -4116 -23426 -8753  
89

**Пример 2**

Массив байт, содержащий значения структуры data:

unsigned char test\_data2[] = {  
 0xbd, 0x52, 0x78, 0xa3, 0x83, 0x62, 0x69, 0xa0,   
 0x50, 0x00, 0x87, 0x0a, 0x70, 0x06, 0x23, 0x59,   
 0x11, 0xf7, 0x01, 0x89, 0x23, 0xb7, 0xa3, 0x38,   
 0x13, 0x6a, 0xe2, 0x8b, 0x3f, 0x11, 0x2d, 0x06,   
 0xa7, 0xf6, 0xfb, 0xba, 0x4e, 0x3c, 0x02, 0xe0,   
 0xbe, 0x23, 0x34, 0x98, 0x2b, 0x9f, 0x5b,   
};

Результат вывода значений полей data на экран:

189  
5942679493002502224  
-8716030957157019145  
1 137 35 183 163 56 19 106 226 139  
16145 11526 -22538 -1094 20028 736 -16861 13464 11167  
91

## Вариант №34

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| unsigned int | 4 | Нет | 2 |
| char | 1 | Нет | 1 |
| short | 4 | Да | 4 |

Порядок байт от старшего к младшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 unsigned int field1;  
 unsigned int field2;  
 char field3[6];  
 short field4[10];  
 char field5[7];  
 char field6;  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для преобразования структуры data в массив байт. В реализации необходимо учесть особенности платформы. Вывести на экран значения массива байт, как показано в примерах ниже:

**Пример 1**

Функция для заполнения полей структуры data:

void test\_code1(struct data \*d) {  
 d->field1 = 1835960635;  
 d->field2 = 3025171430;  
 d->field3[0] = 26;  
 d->field3[1] = 0;  
 d->field3[2] = 29;  
 d->field3[3] = 30;  
 d->field3[4] = 222;  
 d->field3[5] = 243;  
 d->field4[0] = 1042555888;  
 d->field4[1] = 418442451;  
 d->field4[2] = 46145871;  
 d->field4[3] = 1765110607;  
 d->field4[4] = -149217011;  
 d->field4[5] = -891808201;  
 d->field4[6] = -767782621;  
 d->field4[7] = 1749379747;  
 d->field4[8] = 439433817;  
 d->field4[9] = -1775001046;  
 d->field5[0] = 36;  
 d->field5[1] = 145;  
 d->field5[2] = 171;  
 d->field5[3] = 30;  
 d->field5[4] = 83;  
 d->field5[5] = 4;  
 d->field5[6] = 41;  
 d->field6 = 0;  
}

Результат вывода массива байт на экран:

6D 6E 89 3B B4 50 73 E6   
1A 00 1D 1E DE F3 00 00   
3E 24 23 F0 18 F0 EC D3   
02 C0 21 4F 69 35 73 4F   
F7 1B 21 0D CA D8 16 37   
D2 3C 91 23 68 45 6A A3   
1A 31 3A 59 96 33 A2 2A   
24 91 AB 1E 53 04 29 00

**Пример 2**

Функция для заполнения полей структуры data:

void test\_code2(struct data \*d) {  
 d->field1 = 2237777595;  
 d->field2 = 3810998764;  
 d->field3[0] = 44;  
 d->field3[1] = 49;  
 d->field3[2] = 21;  
 d->field3[3] = 134;  
 d->field3[4] = 68;  
 d->field3[5] = 144;  
 d->field4[0] = -1099786405;  
 d->field4[1] = -236935579;  
 d->field4[2] = -1141628905;  
 d->field4[3] = -427090196;  
 d->field4[4] = 1321368253;  
 d->field4[5] = -1594494342;  
 d->field4[6] = -1385311979;  
 d->field4[7] = 1227760944;  
 d->field4[8] = 749860339;  
 d->field4[9] = 345689160;  
 d->field5[0] = 29;  
 d->field5[1] = 35;  
 d->field5[2] = 48;  
 d->field5[3] = 64;  
 d->field5[4] = 42;  
 d->field5[5] = 193;  
 d->field5[6] = 28;  
 d->field6 = 74;  
}

Результат вывода массива байт на экран:

85 61 C6 BB E3 27 39 EC   
2C 31 15 86 44 90 00 00   
BE 72 97 5B F1 E0 A6 65   
BB F4 20 17 E6 8B 1E EC   
4E C2 7A BD A0 F5 F2 7A   
AD 6D D1 15 49 2E 25 30   
2C B1 F5 F3 14 9A CC 48   
1D 23 30 40 2A C1 1C 4A

## Вариант №35

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| int | 4 | Да | 2 |
| float | 4 | Да | 2 |
| char | 1 | Нет | 1 |
| unsigned int | 4 | Нет | 2 |

Порядок байт от старшего к младшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 int field1[3];  
 float field2;  
 char field3;  
 int field4;  
 unsigned int field5;  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для преобразования структуры data в массив байт. В реализации необходимо учесть особенности платформы. Вывести на экран значения массива байт, как показано в примерах ниже:

**Пример 1**

Функция для заполнения полей структуры data:

void test\_code1(struct data \*d) {  
 d->field1[0] = 1601269216;  
 d->field1[1] = 1328002118;  
 d->field1[2] = -1577658960;  
 d->field2 = -0.8323515057563782;  
 d->field3 = 231;  
 d->field4 = 1268857542;  
 d->field5 = 1063588790;  
}

Результат вывода массива байт на экран:

5F 71 6D E0 4F 27 B4 46   
A1 F6 D5 B0 BF 55 14 FD   
E7 00 4B A1 3A C6 3F 65   
13 B6

**Пример 2**

Функция для заполнения полей структуры data:

void test\_code2(struct data \*d) {  
 d->field1[0] = 1296030580;  
 d->field1[1] = -653982289;  
 d->field1[2] = -704471231;  
 d->field2 = -0.35283803939819336;  
 d->field3 = 48;  
 d->field4 = -1663626917;  
 d->field5 = 2521135988;  
}

Результат вывода массива байт на экран:

4D 3F DB 74 D9 05 05 AF   
D6 02 9F 41 BE B4 A7 30   
30 00 9C D7 11 5B 96 45   
7B 74

## Вариант №36

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| long | 8 | Да | 1 |
| char | 1 | Да | 1 |
| int | 4 | Да | 4 |
| unsigned long | 8 | Нет | 1 |

Порядок байт от младшего к старшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 long field1;  
 char field2[2];  
 int field3;  
 char field4;  
 unsigned long field5;  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для преобразования структуры data в массив байт. В реализации необходимо учесть особенности платформы. Вывести на экран значения массива байт, как показано в примерах ниже:

**Пример 1**

Функция для заполнения полей структуры data:

void test\_code1(struct data \*d) {  
 d->field1 = -832303452671527614;  
 d->field2[0] = 83;  
 d->field2[1] = -72;  
 d->field3 = 2024446800;  
 d->field4 = -38;  
 d->field5 = 6563234976880456572;  
}

Результат вывода массива байт на экран:

42 B5 2E 66 6B 10 73 F4   
53 B8 00 00 50 9B AA 78   
DA 7C F3 60 1E 70 4B 15   
5B

**Пример 2**

Функция для заполнения полей структуры data:

void test\_code2(struct data \*d) {  
 d->field1 = 7154924650132250126;  
 d->field2[0] = 30;  
 d->field2[1] = 59;  
 d->field3 = 81591557;  
 d->field4 = -39;  
 d->field5 = 15606232195473503548;  
}

Результат вывода массива байт на экран:

0E 9E D1 95 0F 66 4B 63   
1E 3B 00 00 05 FD DC 04   
D9 3C 75 76 15 7F 79 94   
D8

## Вариант №37

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| unsigned long | 8 | Нет | 8 |
| char | 1 | Нет | 1 |
| float | 4 | Да | 2 |

Порядок байт от младшего к старшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 unsigned long field1;  
 char field2;  
 float field3[10];  
 float field4;  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для разбора массива байт, содержащего структуру data. В реализации необходимо учесть особенности платформы. Вывести на экран значения полей, как показано в примерах ниже.

**Пример 1**

Массив байт, содержащий значения структуры data:

unsigned char test\_data1[] = {  
 0xa5, 0x52, 0x0c, 0x0a, 0xfc, 0xc9, 0x45, 0x26,   
 0x01, 0x00, 0x83, 0xe7, 0x6e, 0xbf, 0x8d, 0x09,   
 0xbe, 0xbe, 0xf8, 0x47, 0x4a, 0xbf, 0x41, 0x24,   
 0x35, 0xbf, 0x52, 0x96, 0x2f, 0xbc, 0x6e, 0x2b,   
 0x00, 0x3e, 0xc0, 0x55, 0x48, 0x3f, 0x01, 0x43,   
 0x33, 0x3f, 0x08, 0x43, 0x46, 0x3f, 0x97, 0x97,   
 0x26, 0xbf, 0x1e, 0x54, 0x3a, 0xbf,   
};

Результат вывода значений полей data на экран:

2757832431171818149  
1  
-0.9332200884819031 -0.3711666166782379 -0.7901606559753418 -0.7075844407081604 -0.010716991499066353 0.1251656711101532 0.7825584411621094 0.7002411484718323 0.7744603157043457 -0.6507505774497986  
-0.7278460264205933

**Пример 2**

Массив байт, содержащий значения структуры data:

unsigned char test\_data2[] = {  
 0xe3, 0xdc, 0xd0, 0x5d, 0x03, 0x5b, 0x0b, 0xd0,   
 0xf8, 0x00, 0xb1, 0x2f, 0x15, 0xbd, 0x66, 0x23,   
 0x89, 0x3d, 0x6d, 0x20, 0x5e, 0x3f, 0x8b, 0x91,   
 0x5d, 0x3f, 0x13, 0xac, 0xce, 0x3d, 0x99, 0x94,   
 0x22, 0x3d, 0x0a, 0x62, 0x92, 0xbe, 0x58, 0xdd,   
 0x09, 0x3f, 0x50, 0x76, 0x23, 0xbd, 0x6d, 0x94,   
 0x65, 0xbf, 0x1a, 0x9d, 0xbd, 0x3e,   
};

Результат вывода значений полей data на экран:

14991175854649826531  
248  
-0.03642243519425392 0.06696204841136932 0.8676822781562805 0.8655020594596863 0.10091414302587509 0.03969249501824379 -0.2859042286872864 0.5385336875915527 -0.039907753467559814 -0.8967960476875305  
0.37033921480178833

## Вариант №38

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| unsigned int | 2 | Нет | 4 |
| char | 1 | Да | 1 |
| int | 2 | Да | 4 |

Порядок байт от младшего к старшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 unsigned int field1;  
 unsigned int field2[6];  
 char field3;  
 int field4;  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для разбора массива байт, содержащего структуру data. В реализации необходимо учесть особенности платформы. Вывести на экран значения полей, как показано в примерах ниже.

**Пример 1**

Массив байт, содержащий значения структуры data:

unsigned char test\_data1[] = {  
 0xc1, 0x18, 0x00, 0x00, 0xc0, 0x08, 0x00, 0x00,   
 0x64, 0x49, 0x00, 0x00, 0x4b, 0x06, 0x00, 0x00,   
 0xcb, 0x9e, 0x00, 0x00, 0xd3, 0x8d, 0x00, 0x00,   
 0xca, 0xb7, 0x85, 0x00, 0x01, 0xd8,   
};

Результат вывода значений полей data на экран:

6337  
2240 18788 1611 40651 36307 47050  
-123  
-10239

**Пример 2**

Массив байт, содержащий значения структуры data:

unsigned char test\_data2[] = {  
 0xb9, 0xb4, 0x00, 0x00, 0x9c, 0x57, 0x00, 0x00,   
 0xc3, 0x4d, 0x00, 0x00, 0x8e, 0x5e, 0x00, 0x00,   
 0x26, 0xe0, 0x00, 0x00, 0x7f, 0x63, 0x00, 0x00,   
 0x7d, 0x9d, 0xcb, 0x00, 0xe5, 0x44,   
};

Результат вывода значений полей data на экран:

46265  
22428 19907 24206 57382 25471 40317  
-53  
17637

## Вариант №39

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| char | 1 | Да | 1 |
| int | 2 | Да | 1 |
| long | 4 | Да | 8 |

Порядок байт от старшего к младшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 char field1[7];  
 int field2;  
 long field3;  
 char field4;  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для разбора массива байт, содержащего структуру data. В реализации необходимо учесть особенности платформы. Вывести на экран значения полей, как показано в примерах ниже.

**Пример 1**

Массив байт, содержащий значения структуры data:

unsigned char test\_data1[] = {  
 0x3c, 0xfd, 0xca, 0x76, 0x7c, 0x52, 0xa9, 0xbb,   
 0x51, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   
 0xd1, 0x22, 0x38, 0xf6, 0x07,   
};

Результат вывода значений полей data на экран:

60 -3 -54 118 124 82 -87  
-17583  
-786286346  
7

**Пример 2**

Массив байт, содержащий значения структуры data:

unsigned char test\_data2[] = {  
 0x48, 0xfa, 0x6c, 0x64, 0x1d, 0x13, 0x28, 0xf5,   
 0x42, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,   
 0x2e, 0xcc, 0x6e, 0xb6, 0x22,   
};

Результат вывода значений полей data на экран:

72 -6 108 100 29 19 40  
-2750  
785149622  
34

## Вариант №40

Написать программу на C/C++ для работы со структурой данных c учетом архитектурных особенностей некоторой платформы.

Информация о платформе представлена в таблице:

| Тип | Размер (байт) | Знак | Выравнивание (байт) |
| --- | --- | --- | --- |
| char | 1 | Нет | 1 |
| int | 4 | Да | 8 |
| long long | 8 | Да | 2 |
| double | 8 | Да | 2 |

Порядок байт от младшего к старшему.

Структура данных на целевой платформе имеет следующий вид:

struct data {  
 char field1[5];  
 int field2[2];  
 long long field3[3];  
 int field4;  
 char field5;  
 double field6;  
};

Типы в приведенной структуре данных, возможно, придется адаптировать к используемому в решении задачи компилятору.

Написать функцию для разбора массива байт, содержащего структуру data. В реализации необходимо учесть особенности платформы. Вывести на экран значения полей, как показано в примерах ниже.

**Пример 1**

Массив байт, содержащий значения структуры data:

unsigned char test\_data1[] = {  
 0x14, 0x78, 0x59, 0x10, 0xab, 0x00, 0x00, 0x00,   
 0x52, 0x39, 0xb3, 0x15, 0x00, 0x00, 0x00, 0x00,   
 0xf4, 0xc0, 0xd6, 0xfc, 0x67, 0xea, 0x78, 0xc4,   
 0x5b, 0x52, 0x5e, 0xba, 0x9a, 0xb2, 0x36, 0x1a,   
 0x43, 0xf7, 0x7b, 0x58, 0x62, 0x06, 0xa2, 0x60,   
 0x88, 0x47, 0xb1, 0xba, 0x00, 0x00, 0x00, 0x00,   
 0xed, 0x05, 0xf8, 0xcf, 0xef, 0x00, 0xac, 0xdb,   
 0x1e, 0xab, 0x41, 0x78, 0xd4, 0xbf,   
};

Результат вывода значений полей data на экран:

20 120 89 16 171  
364067154 -53034764  
-5017482380752393625 6375961565047730842 -4994131860714813854  
-805829139  
239  
-0.31983987532606295

**Пример 2**

Массив байт, содержащий значения структуры data:

unsigned char test\_data2[] = {  
 0xe8, 0x67, 0x05, 0x53, 0x36, 0x00, 0x00, 0x00,   
 0xa2, 0x6f, 0xe3, 0x45, 0x00, 0x00, 0x00, 0x00,   
 0x20, 0x74, 0x82, 0x2d, 0xcb, 0x91, 0x5c, 0x9a,   
 0x00, 0x66, 0x25, 0xd3, 0x8b, 0xea, 0x38, 0x68,   
 0xc6, 0xe9, 0xb7, 0x9b, 0xad, 0x1f, 0x8a, 0xdc,   
 0xbe, 0x89, 0x7b, 0x19, 0x00, 0x00, 0x00, 0x00,   
 0x6d, 0x01, 0xf3, 0xdb, 0x34, 0x00, 0x88, 0xbc,   
 0x53, 0x07, 0x09, 0x05, 0xe0, 0xbf,   
};

Результат вывода значений полей data на экран:

232 103 5 83 54  
1172533154 763524128  
-3232065004792671797 -7226050038731314549 1836212725920440237  
-604831379  
52  
-0.5006146567447027