Задание №3.4 в рамках вычислительного практикума. Представление в памяти структур и объединений

Локальные переменные

Исходный код с инициализированными локальными переменными:

```
int main(void)
{
    int a = -54;
    double d = 31.42;
    char c = 'a';
    short q = 32;

    return 0;
}
```

Дамп памяти для переменной a

```
(gdb) x/4xb &a
0x5ffe9c:    0xca   0xff   0xff   0xff
```

Дамп памяти для переменной d

```
(gdb) x/8xb &d
0x5ffe90:    0xec   0x51   0xb8   0x1e   0x85   0x6b   0x3f   0x40
```

Дамп памяти для переменной c

```
(gdb) x/1xb &c
0x5ffe8f:    0x61
```

Дамп памяти для переменной q

```
(gdb) x/2xb &q
0x5ffe8c:    0x20   0x00
```

Исходя из дампов видно, что переменная ‘q’ встречается раньше всех, а переменная ‘a’ позже всех. Чтобы узнать, Все локальные переменные занимают (9c + 4) – (8c) = 20 байт

Дамп памяти, который содержит все локальные переменные:

```
(gdb) x/20xb &q
0x5ffe8c:	0x20	0x00	0x00	0x61	0xec	0x51	0xb8	0x1e
0x5ffe94:	0x85	0x6b	0x3f	0x40	0x60	0x46	0xa6	0x00
0x5ffe9c:	0xca	0xff	0xff	0xff
```

Сведения о переменных:

Имя	Размер (байт)	Адрес
a	4	0x5ffe9c
d	8	0x5ffe90
c	1	0x5ffe8f
q	2	0x5ffe8c

Выводы: Переменные в памяти хранятся друг за другом, и каждая переменная находится по адресу, кратному её размеру.

Структуры

Описание структуры и структурной переменной (переменная должна быть проинициализирована)

```
struct Person
{
   int age;
   char genger;
   double height;
};

int main(void)
{
   struct Person person = {25, 'M', 175.5};

   return 0;
}
```

Дамп памяти структурной переменной:

```
(gdb) p sizeof(person)
$1 = 16
(gdb) x/16xb &person
0x5ffe90:	0x19	0x00	0x00	0x00	0x4d	0x00	0x00	0x00
```

```
0x5ffe98:	0x00	0x00	0x00	0x00	0x00	0xf0	0x65	0x40
```

Сведения о полях структурной переменной:

Имя	Размер (байт)	Адрес
age	4	0x5ffe90
genger	1	0x5ffe94
height	8	0x5ffe98

Адрес поля должен быть кратен размеру поля

Переменная структурного типа располагается по адресу, кратному её первому полю

Описание упакованной структуры

```
#pragma pack(push, 1)
struct Person
{
   int age;
   char genger;
   double height;
};
#pragma pack(pop)

int main(void)
{
   struct Person person = {25, 'M', 175.5};

   return 0;
}
```

Дамп памяти упакованной структурной переменной:

```
(gdb) p sizeof(person)
$1 = 13
(gdb) x/13xb &person
0x5ffe93:	0x19	0x00	0x00	0x00	0x4d	0x00	0x00	0x00
0x5ffe9b:	0x00	0x00	0xf0	0x65	0x40
```

Сведения о полях упакованной структурной переменной:

Имя	Размер (байт)	Адрес
age	4	0x5ffe93

genger	1	0x5ffe97
height	8	0x5ffe98

В моем случае независимо от расположения полей структуры, она всегда занимает 16 байт.

В моем случае завершающего выравнивания нет, так последние байты памяти в дампе используются

Объединения

Описание объединения и инициализации одного из его полей

```
union Person
{
   int age;
   char genger;
   double height;
};

int main(void)
{
   union Person person;
   person.age = 25;

   return 0;
}
```

Дамп памяти объединения (поле age имеет значение 25)

```
(gdb) p sizeof(&person)
$1 = 8
(gdb) x/8xb &person
0x5ffe98:      0x19    0x00    0x00    0x00    0x00    0x00    0x00    0x00
```

Дамп памяти объединения (поле genger имеет значение ‘M’)

```
(gdb) x/8xb &person
0x5ffe98:      0x4d    0x46    0x78    0x00    0x00    0x00    0x00    0x00
```

Дамп памяти объединения (поле height имеет значение 175.5)

```
(gdb) x/8xb &person
0x5ffe98:      0x00    0x00    0x00    0x00    0x00    0xf0    0x65    0x40
```