Балтийский государственный технический университет  
«ВОЕНМЕХ» им. Д. Ф. Устинова

Кафедра О7 «Информационные системы и программная инженерия»

**Практическая работа №3**по дисциплине «Информатика: Основы программирования»  
на тему «Указатели»

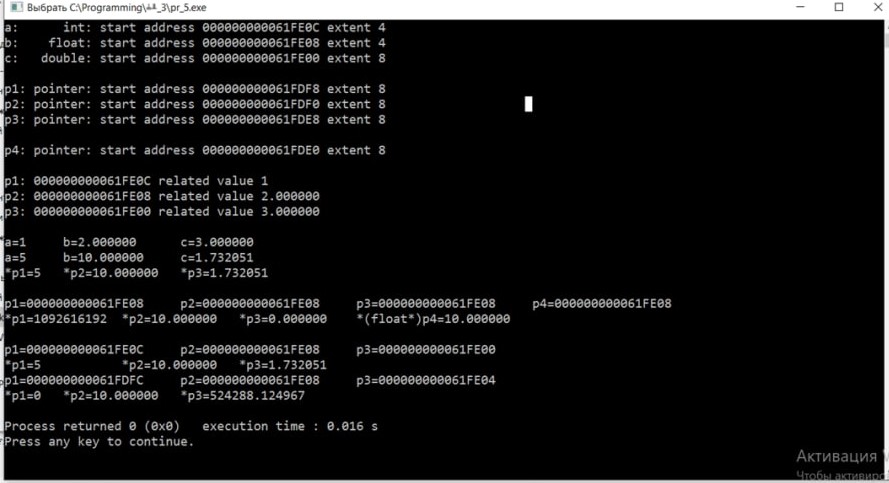
Выполнил:  
Студент Вяткин Н. А.  
Группа О722Б  
  
Преподаватель: Назарова М. А.

Санкт-Петербург  
2022 г.

**Задание 1.**

Проанализировать текст представленной программы и выдаваемые программой результаты. Объяснить, почему результаты именно такие.

Результаты работы программы:



Текст программы:

#include<stdlib.h>

#include<stdio.h>

#include<math.h>

int main()

{

/\* «Обычные» переменные \*/

int a = 1;

float b = 2;

double c = 3;

/\* Указатели \*/

int \*p1 = &a;

float \*p2 = &b;

double \*p3 = &c;

void \*p4 = NULL;

/\* Адреса «обычных» переменных и размер выделяемой памяти \*/

printf("a: int: start address %p extent %d\n",&a,sizeof(a));

printf("b: float: start address %p extent %d\n",&b,sizeof(b));

printf("c: double: start address %p extent %d\n\n",&c,sizeof(c));

/\* Адреса указателей и размер выделяемой памяти \*/

printf("p1: pointer: start address %p extent %d\n",&p1,sizeof(p1));

printf("p2: pointer: start address %p extent %d\n",&p2,sizeof(p2));

printf("p3: pointer: start address %p extent %d\n\n",&p3,sizeof(p3));

printf("p4: pointer: start address %p extent %d\n\n",&p4,sizeof(p4));

/\* Значения, на которые ссылаются указатели \*/

printf("p1: %p related value %d\n",p1,\*p1);

printf("p2: %p related value %f\n",p2,\*p2);

printf("p3: %p related value %lf\n\n",p3,\*p3);

Размещение переменных в памяти

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 000000000061FDE0 | 000000000061FDE1 | 000000000061FDE2 | 000000000061FDE3 | 000000000061FDE4 | 000000000061FDE5 | 000000000061FDE6 | 000000000061FDE7 | 000000000061FDE8 | 000000000061FDE9 | 000000000061FDEA | 000000000061FDEB | 000000000061FDEC | 000000000061FDED | 000000000061FDEE | 000000000061FDEF | 000000000061FDF0 | 000000000061FDF1 | 000000000061FDF2 | 000000000061FDF3 | 000000000061FDF4 | 000000000061FDF5 | 000000000061FDF6 | 000000000061FDF7 | 000000000061FDF8 | 000000000061FDF9 | 000000000061FDFA | 000000000061FDFB | 000000000061FDFC | 000000000061FDFD | 000000000061FDFE | 000000000061FDFF | 000000000061FE00 | 000000000061FE01 | 000000000061FE02 | 000000000061FE03 | 000000000061FE04 | 000000000061FE05 | 000000000061FE06 | 000000000061FE07 | 000000000061FE08 | 000000000061FE09 | 000000000061FE0A | 000000000061FE0B | 000000000061FE0C | 000000000061FE0D | 000000000061FE0E | 000000000061FE0F |
| переменные | p4 | | | | | | | | p3 | | | | | | | | p2 | | | | | | | | p1 | | | | | | | | c | | | | | | | | b | | | | a | | | |
| значения переменных | NULL | | | | | | | | 000000000061FE00 | | | | | | | | 000000000061FE08 | | | | | | | | 000000000061FE0C | | | | | | | | 3.000000 | | | | | | | | 2.000000 | | | | 1 | | | |
| адресуемые области указателей |  | | | | | | | |  | | | | | | | |  | | | | | | | |  | | | | | | | | p3(000000000061FE00) | | | | | | | | p2(000000000061FE08) | | | | p1(000000000061FE0C) | | | |
| значения в адресуемых областях |  | | | | | | | |  | | | | | | | |  | | | | | | | |  | | | | | | | | 3.000000 | | | | | | | | 2.000000 | | | | 1 | | | |

Вывод: Под указатели выделяется по 8 байт памяти, в указатели заключается адрес переменной, на которую указывает указатель, указатель p1=адресу переменной a, указатель p2=адресу переменной b, указатель p3=адресу переменной c, поэтому \*p1=a (тип int), \*p2=b (тип float), \*p3=c, p4=NULL-нулевой адрес, под переменные a, b, c отводится то количество байт, которое заложено их типом. Указатели представляются также как переменная, на которую они указывают (a=1(int) \*p1=1, b=2.000000(float) \*p2=2.000000, \*p3=3.000000 c=3.000000(double))

/\* Использование указателей в выражениях \*/

printf("a=%d\tb=%f\tc=%lf\n",a,b,c);

\*p1 = 5;

\*p2 = \*p2 \* \*p1;

\*p3 = sqrt(\*p3);

printf("a=%d\tb=%f\tc=%lf\n",a,b,c);

printf("\*p1=%d\t\*p2=%f\t\*p3=%lf\n\n",\*p1,\*p2,\*p3);

/\* Присваивание указателей \*/

p1 = (int\*)p2;

p3 = (double\*)p2;

p4 = p2;

printf("p1=%p\tp2=%p\tp3=%p\tp4=%p\n",p1,p2,p3,p4);

printf("\*p1=%d\t\*p2=%f\t\*p3=%f\t\*(float\*)p4=%f\n\n",\*p1,\*p2,\*p3,\*(float\*)p4);

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 000000000061FDE0 | 000000000061FDE1 | 000000000061FDE2 | 000000000061FDE3 | 000000000061FDE4 | 000000000061FDE5 | 000000000061FDE6 | 000000000061FDE7 | 000000000061FDE8 | 000000000061FDE9 | 000000000061FDEA | 000000000061FDEB | 000000000061FDEC | 000000000061FDED | 000000000061FDEE | 000000000061FDEF | 000000000061FDF0 | 000000000061FDF1 | 000000000061FDF2 | 000000000061FDF3 | 000000000061FDF4 | 000000000061FDF5 | 000000000061FDF6 | 000000000061FDF7 | 000000000061FDF8 | 000000000061FDF9 | 000000000061FDFA | 000000000061FDFB | 000000000061FDFC | 000000000061FDFD | 000000000061FDFE | 000000000061FDFF | 000000000061FE00 | 000000000061FE01 | 000000000061FE02 | 000000000061FE03 | 000000000061FE04 | 000000000061FE05 | 000000000061FE06 | 000000000061FE07 | 000000000061FE08 | 000000000061FE09 | 000000000061FE0A | 000000000061FE0B | 000000000061FE0C | 000000000061FE0D | 000000000061FE0E | 000000000061FE0F |
| переменные | p4 | | | | | | | | p3 | | | | | | | | p2 | | | | | | | | p1 | | | | | | | | c | | | | | | | | b | | | | a | | | |
| значения переменных | 000000000061FE08 | | | | | | | | 000000000061FE08 | | | | | | | | 000000000061FE08 | | | | | | | | 000000000061FE08 | | | | | | | | 1.732051 | | | | | | | | 10.000000 | | | | 5 | | | |
| адресуемые области указателей |  | | | | | | | |  | | | | | | | |  | | | | | | | |  | | | | | | | |  | | | | | | | | p2(000000000061FE08) | | | |  | | | |
| адресуемые области указателей |  | | | | | | | |  | | | | | | | |  | | | | | | | |  | | | | | | | |  | | | | | | | | p3(000000000061FE08) | | | | | | | |
| адресуемые области указателей |  | | | | | | | |  | | | | | | | |  | | | | | | | |  | | | | | | | |  | | | | | | | | p1(000000000061FE08) | | | |  | | | |
| значения в адресуемых областях |  | | | | | | | |  | | | | | | | |  | | | | | | | |  | | | | | | | |  | | | | | | | | 10.000000 | | | |  | | | |
| значения в адресуемых областях |  | | | | | | | |  | | | | | | | |  | | | | | | | |  | | | | | | | |  | | | | | | | | 0.000000 | | | | | | | |
| значения в адресуемых областях |  | | | | | | | |  | | | | | | | |  | | | | | | | |  | | | | | | | |  | | | | | | | | 1092616192 | | | |  | | | |

Вывод: \*p1=5, следовательно, в адрес переменной a присвоили новое значение a=5(изменили значение которое находится по адресу указателя p1) ;

\*p2=\*p1\*\*p2, следовательно, в адрес переменной b присвоили новое значение b=a\*b=5\*2=10.000000(так как происходят действия с значениями адресов указателей, содержащими значение переменных a и b, также происходит неявное приведение типов);

\*p3=sqrt(\*p3), следовательно, в адрес переменной c присвоили новое значение c=sqrt(c)(то есть мы изменяем значение переменной c, изменяя значение которое находится по адресу указателя)

При помощи \* мы можем вытаскивать значение по адресу указателя и действовать с ним, как с переменной. (операция разыменовывания)

При присвоение указателю указателя другого типа необходимо использовать приведение типов;

После присвоения указателю p1=(int\*)p2- указатель p1 стал равен новому адресу (000000000061FE08) и при разыменовывание значения по этому адресу мы отличное число от того, которое мы получим по адресу p2 при разыменовывание, это связано с тем что значение по адресу в типе int будет иметь другое представление по сравнению с представлением значения по тому же адресу в типе float.(10 в типе float = 01000001001000000000000000000000, при переводе этого числа в int мы получим = 1092616192)

После присвоения указателю p3=(double \*)p2 указатель p3 стал равен новому адресу(000000000061FE08) и при вытаскивание значения по этому адресу мы получим отличное число от того, которое мы вытащим по адресу p2, это связано с тем что значение по адресу в типе double будет иметь другое представление по сравнению с представлением значения по тому же адресу в типе float.

При присвоении указателю пустого типа адрес типа float(p4=p2) адрес пустого типа станет адресом типа float, поэтому при разыменовывании мы получим тоже самое значение что и значение по адресу p2.

/\* Изменение значений указателей \*/

p1++;

p3--;

printf("p1=%p\tp2=%p\tp3=%p\n",p1,p2,p3);

printf("\*p1=%d\t\t\*p2=%f\t\*p3=%lf\n",\*p1,\*p2,\*p3);

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 000000000061FDE0 | 000000000061FDE1 | 000000000061FDE2 | 000000000061FDE3 | 000000000061FDE4 | 000000000061FDE5 | 000000000061FDE6 | 000000000061FDE7 | 000000000061FDE8 | 000000000061FDE9 | 000000000061FDEA | 000000000061FDEB | 000000000061FDEC | 000000000061FDED | 000000000061FDEE | 000000000061FDEF | 000000000061FDF0 | 000000000061FDF1 | 000000000061FDF2 | 000000000061FDF3 | 000000000061FDF4 | 000000000061FDF5 | 000000000061FDF6 | 000000000061FDF7 | 000000000061FDF8 | 000000000061FDF9 | 000000000061FDFA | 000000000061FDFB | 000000000061FDFC | 000000000061FDFD | 000000000061FDFE | 000000000061FDFF | 000000000061FE00 | 000000000061FE01 | 000000000061FE02 | 000000000061FE03 | 000000000061FE04 | 000000000061FE05 | 000000000061FE06 | 000000000061FE07 | 000000000061FE08 | 000000000061FE09 | 000000000061FE0A | 000000000061FE0B | 000000000061FE0C | 000000000061FE0D | 000000000061FE0E | 000000000061FE0F |
| переменные | p4 | | | | | | | | p3 | | | | | | | | p2 | | | | | | | | p1 | | | | | | | | c | | | | | | | | b | | | | a | | | |
| значения переменных | NULL | | | | | | | | 000000000061FE00 | | | | | | | | 000000000061FE08 | | | | | | | | 000000000061FE0C | | | | | | | | 1.732051 | | | | | | | | 10.000000 | | | | 5 | | | |
| адресуемые области указателей |  | | | | | | | |  | | | | | | | |  | | | | | | | |  | | | | | | | | p3(000000000061FE00) | | | | | | | | p2(000000000061FE08) | | | | p1(000000000061FE0C) | | | |
| значения в адресуемых областях |  | | | | | | | |  | | | | | | | |  | | | | | | | |  | | | | | | | | 1.732051 | | | | | | | | 10.000000 | | | | 5 | | | |

Вывод: При прибавлении 1 и вычитание 1 к указателям мы получили изначальные адреса переменных a и c, то есть указатели стали равны \*p1=5, \*p2=1.732051

Арифметические действия с указателями происходят как смещение вперед или назад на n количество разрядов (n\*объем выделяемой под указатель памяти(sizeof())(зависит от типа переменной, на которую указывает указатель): Например, p1++ это значит, что n=1, следовательно, 000000000061FE08+1\*4(sizeof(p1))= 000000000061FE0C(так как p1 указатель на int) ; p3-- значит, что n=1, следовательно, 000000000061FE08-1\*8(sizeof(p3))= 000000000061FE00;(так как p3 указатель на double)

p1 -= 4;

p3 = (double\*)&a - 1;

printf("p1=%p\tp2=%p\tp3=%p\n",p1,p2,p3);

printf("\*p1=%d\t\*p2=%f\t\*p3=%lf\n",\*p1,\*p2,\*p3);

return 0;

}

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 000000000061FDE0 | 000000000061FDE1 | 000000000061FDE2 | 000000000061FDE3 | 000000000061FDE4 | 000000000061FDE5 | 000000000061FDE6 | 000000000061FDE7 | 000000000061FDE8 | 000000000061FDE9 | 000000000061FDEA | 000000000061FDEB | 000000000061FDEC | 000000000061FDED | 000000000061FDEE | 000000000061FDEF | 000000000061FDF0 | 000000000061FDF1 | 000000000061FDF2 | 000000000061FDF3 | 000000000061FDF4 | 000000000061FDF5 | 000000000061FDF6 | 000000000061FDF7 | 000000000061FDF8 | 000000000061FDF9 | 000000000061FDFA | 000000000061FDFB | 000000000061FDFC | 000000000061FDFD | 000000000061FDFE | 000000000061FDFF | 000000000061FE00 | 000000000061FE01 | 000000000061FE02 | 000000000061FE03 | 000000000061FE04 | 000000000061FE05 | 000000000061FE06 | 000000000061FE07 | 000000000061FE08 | 000000000061FE09 | 000000000061FE0A | 000000000061FE0B | 000000000061FE0C | 000000000061FE0D | 000000000061FE0E | 000000000061FE0F |
| переменные | p4 | | | | | | | | p3 | | | | | | | | p2 | | | | | | | | p1 | | | | | | | | c | | | | | | | | b | | | | a | | | |
| значения переменных | NULL | | | | | | | | 00000000061FE04 | | | | | | | | 000000000061FE08 | | | | | | | | 000000000061FDFC | | | | | | | | 1.732051 | | | | | | | | 10.000000 | | | | 5 | | | |
| адресуемые области указателей |  | | | | | | | |  | | | | | | | |  | | | | | | | |  | | | | p1(000000000061FDFC) | | | |  | | | |  | | | | p2(000000000061FE08) | | | |  | | | |
| значения в адресуемых областях |  | | | | | | | |  | | | | | | | |  | | | | | | | |  | | | | 0 | | | |  | | | |  | | | | 10.000000 | | | |  | | | |
| адресуемые области указателей |  | | | | | | | |  | | | | | | | |  | | | | | | | |  | | | | | | | |  | | | | p3(000000000061FE04) | | | | | | | |  | | | |
| значения в адресуемых областях |  | | | | | | | |  | | | | | | | |  | | | | | | | |  | | | | | | | |  | | | | 524288.124967 | | | | | | | |  | | | |

При изменении значения указателя на k он сдвигается к k-му следующему числу данного типа, то есть для указателей на целые числа на k\*sizeof(тип) байт- p1-=4 значит, что 000000000061FE0C-4\*4=000000000061FDFC, при присвоение указателю p3 адреса переменной a типа int при помощи явного приведения типов и после вычитания 1 мы получаем, что адрес p3=(double\*)&a-1 = 000000000061FE0C-1\*8=000000000061FE04, когда мы пытаемся вытащить значения по адресам p1 и p3 программа выводит хаотичные числа, так как по данным адресам не хранятся никакие объявленные переменные.

*Выводы*: Под переменные отводится количество байт памяти заложенное ее типом, под указатели отводится заданное ОС количество байт памяти(8). Указатели хранят адреса переменных, по адресу можно вытащить значение переменной(разыменование). Изменять значение переменных можно с помощью указателей, то есть использовать и изменять значение которое находится по адресу указателя. При присваивании указателю указатель используется явное приведение типов, если их типы различны, то значение которое находится по адресу, который получил указатель после присвоения указателя другого типа, будет отличаться, так как в разных типах значения представляются по-разному. При изменении значения указателя на k он сдвигается к k-му следующему числу данного типа, то есть для указателей на целые числа на k\*sizeof(тип) байт. NULL – нулевой адрес, не соответствующий никакому реальному адресу; при помощи явного приведения типов указателю типа void можно присвоить указатель любого типа.

**Задание 2.**

Операционная система Windows 10, среда разработки Code::Blocks

Проанализировать текст представленной программы, найти в нем синтаксические ошибки и исправить их, в начало программы добавить вывод на экран адресов всех переменных, а в конец – значений всех переменных, проанализировать полученные результаты и объяснить, почему они именно такие. Заменить инструкцию «m+=2;» инструкцией «m++;», проанализировать результат.

Текст измененной программы:

Результат работы программ:

#include <stdio.h>

#include <stdlib.h>

int main()

{

char \*p, c;

int \*a, b;

float \*x, y = 3.5;

double \*m, n;

printf("char:\tadress \*p = %p\tadress c = %p\nint:\tadress \*a = %p\tadress b = %p\nfloat:\tadress \*x = %p\tadress y = %p\ndouble:\tadress \*m = %p\tadress n = %p\n",&p,&c,&a,&b,&x,&y,&m,&n);

a = &b;

printf("Enter b = ");

scanf("%d", a);

printf("a=%p\t\*a=%d\tb=%d\n", a, \*a, b);

p =(char\*)a;

c = \*p;

\*p = \*(p+3);

\*(p+3) = c;

printf("p=%p\tc=%d\ta=%p\tb=%d\n", p, c, a, b);

x = &y;

printf("x=%p\t\*x=%f\ty=%f\n", x, \*x, y);

a =(int\*) x;

\*a = \*x;

printf("a=%p\t\*a=%d\tx=%p\t\*x=%f\ty=%f\n", a, \*a, x, \*x, y);

a = &b;

y = 12345.6789;

printf("x=%p\t\*x=%f\ty=%f\n", x, \*x, y);

p =(char\*)x;

c = \*p;

\*p = \*(p+3);

\*(p+3) = c;

printf("p=%p\tc=%d\tx=%p\ty=%f\n", p, c, x, y);

m = &n;

printf("m=%p\t\*m=%lf\tn=%lf\n", m, \*m, n);

n = 5.5;

printf("m=%p\t\*m=%lf\tn=%lf\n", m, \*m, n);

b = n = y = 1.7;

printf("b=%d\ty=%f\tn=%lf\n", b, y, n);

printf("\*a=%d\t\*x=%f\t\*m=%lf\n", \*a, \*x, \*m);

m+=2;

printf("n=%lf\tn=%p\tm=%p\n", n, &n, m);

\*m = (float)\*a - n + (int)\*x;

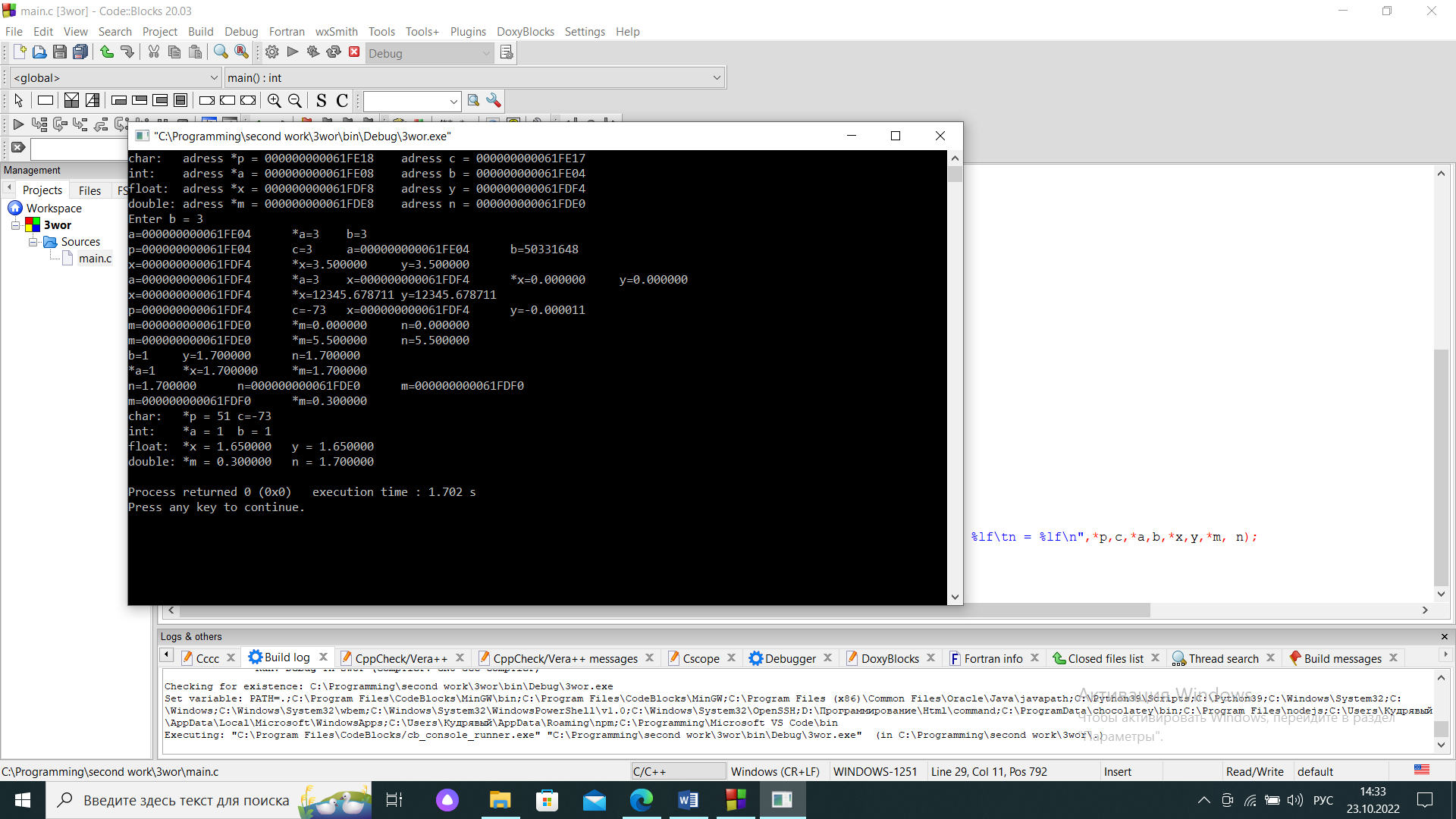
printf("m=%p\t\*m=%lf\n", m, \*m);

printf("char:\t\*p = %d\tc=%d\nint:\t\*a = %d\tb = %d\nfloat:\t\*x = %f\ty = %f\ndouble:\t\*m = %lf\tn = %lf\n",\*p,c,\*a,b,\*x,y,\*m, n);

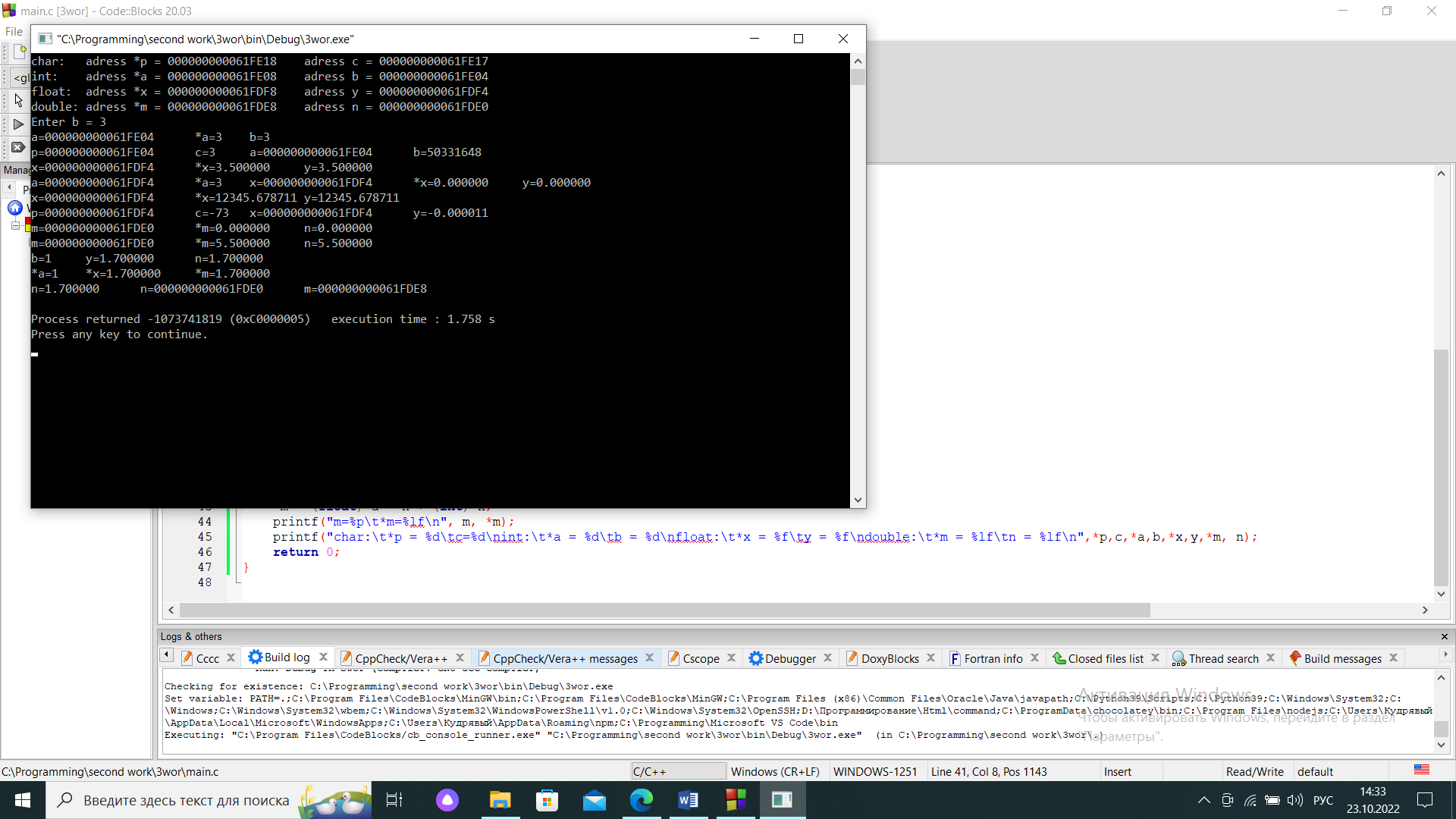
return 0;

}

первый вариант (m+=2;)



второй вариант (m++;)



Размещение переменных в памяти

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 000000000061FDE0 | 000000000061FDE1 | 000000000061FDE2 | 000000000061FDE3 | 000000000061FDE4 | 000000000061FDE5 | 000000000061FDE6 | 000000000061FDE7 | 000000000061FDE8 | 000000000061FDE9 | 000000000061FDEA | 000000000061FDEB | 000000000061FDEC | 000000000061FDED | 000000000061FDEE | 000000000061FDEF | 000000000061FDF0 | 000000000061FDF1 | 000000000061FDF2 | 000000000061FDF3 | 000000000061FDF4 | 000000000061FDF5 | 000000000061FDF6 | 000000000061FDF7 | 000000000061FDF8 | 000000000061FDF9 | 000000000061FDFA | 000000000061FDFB | 000000000061FDFC | 000000000061FDFD | 000000000061FDFE | 000000000061FDFF | 000000000061FE00 | 000000000061FE01 | 000000000061FE02 | 000000000061FE03 | 000000000061FE04 | 000000000061FE05 | 000000000061FE06 | 000000000061FE07 | 000000000061FE08 | 000000000061FE09 | 000000000061FE0A | 000000000061FE0B | 000000000061FE0C | 000000000061FE0D | 000000000061FE0E | 000000000061FE0F | 000000000061FE10 | 000000000061FE11 | 000000000061FE12 | 000000000061FE13 | (000000000061FE14) (000000000061FE15) (000000000061FE16) | 000000000061FE17 | | 000000000061FE18 | 000000000061FE19 | 000000000061FE1A | 000000000061FE1B | 000000000061FE1C | 000000000061FE1D | 000000000061FE1E | 000000000061FE1F |
| переменные | n | | | | | | | | m | | | | | | | |  |  |  |  | y | | | | x | | | | | | | |  |  |  |  | b | | | | a | | | | | | | |  |  |  |  |  | c | | p | | | | | | | |
| значения переменных |  | | | | | | | |  | | | | | | | |  |  |  |  | 3.5 | | | |  | | | | | | | |  |  |  |  | 3 | | | | 000000000061FE04 | | | | | | | |  |  |  |  |  |  | |  | | | | | | | |
| адресуемые области указателей |  | | | | | | | |  | | | | | | | |  |  |  |  |  | | | |  | | | | | | | |  |  |  |  | a | | | |  | | | | | | | |  |  |  |  |  |  | |  | | | | | | | |
| значения в адресуемых областях |  | | | | | | | |  | | | | | | | |  |  |  |  |  | | | |  | | | | | | | |  |  |  |  | 3 | | | |  | | | | | | | |  |  |  |  |  |  |  | | | | | | | | |

p =(char\*)a;/\*присваиваем указателю p адрес переменной b(a - указатель на переменную b)p теперь тоже указывает на переменную b\*/

c = \*p;/\*присваиваем переменной c типа char значение, которое находится по адресу указателя p, то есть присваиваем переменной c младший разряд переменной b - 4 байт c= 00000011=3 \*/

\*p = \*(p+3); /\* указателю p типа char\* на переменную b тип int (при разыменовывании равен 4 байту переменной b) присваиваем указатель (p+3) который указывает на 1 байт переменной b(так как обратный порядок следования байт), этим действием мы меняем изменяем четвертый байт на первый байт в переменной b(00000011= 00000000), теперь \*p= 00000000(4 байт)(обратный порядок следования байт) \*/

\*(p+3) = c;/\*меняем местами первый и четвертый байт в переменной b с помощью указателя типа char\* и переменной типа char со значением 4 байта, (\*(p+3)=c(00000000=00000011)), поэтому b=00000011000000000000000000000000=50331648(обратный порядок следования байт)\*/

printf("p=%p\tc=%d\ta=%p\tb=%d\n", p, c, a, b);

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 000000000061FDE0 | 000000000061FDE1 | 000000000061FDE2 | 000000000061FDE3 | 000000000061FDE4 | 000000000061FDE5 | 000000000061FDE6 | 000000000061FDE7 | 000000000061FDE8 | 000000000061FDE9 | 000000000061FDEA | 000000000061FDEB | 000000000061FDEC | 000000000061FDED | 000000000061FDEE | 000000000061FDEF | 000000000061FDF0 | 000000000061FDF1 | 000000000061FDF2 | 000000000061FDF3 | 000000000061FDF4 | 000000000061FDF5 | 000000000061FDF6 | 000000000061FDF7 | 000000000061FDF8 | 000000000061FDF9 | 000000000061FDFA | 000000000061FDFB | 000000000061FDFC | 000000000061FDFD | 000000000061FDFE | 000000000061FDFF | 000000000061FE00 | 000000000061FE01 | 000000000061FE02 | 000000000061FE03 | 000000000061FE04 | 000000000061FE05 | 000000000061FE06 | 000000000061FE07 | | 000000000061FE08 | 000000000061FE09 | 000000000061FE0A | 000000000061FE0B | 000000000061FE0C | 000000000061FE0D | 000000000061FE0E | 000000000061FE0F | 000000000061FE10 | 000000000061FE11 | 000000000061FE12 | 000000000061FE13 | (000000000061FE14) (000000000061FE15) (000000000061FE16) | 000000000061FE17 | | 000000000061FE18 | 000000000061FE19 | 000000000061FE1A | 000000000061FE1B | 000000000061FE1C | 000000000061FE1D | 000000000061FE1E | 000000000061FE1F |
| переменные | n | | | | | | | | m | | | | | | | |  |  |  |  | y | | | | x | | | | | | | |  |  |  |  | b | | | | | a | | | | | | | |  |  |  |  |  | c | | p | | | | | | | |
| значения переменных |  | | | | | | | |  | | | | | | | |  |  |  |  | 3,5 | | | | 000000000061FDF4 | | | | | | | |  |  |  |  | 3 | | | | | 000000000061FE04 | | | | | | | |  |  |  |  |  | 3 | | 000000000061FE04 | | | | | | | |
| адресуемые области указателей |  | | | | | | | |  | | | | | | | |  |  |  |  |  | | | |  | | | | | | | |  |  |  |  | a | | | | |  | | | | | | | |  |  |  |  |  |  | |  | | | | | | | |
| значения в адресуемых областях |  | | | | | | | |  | | | | | | | |  |  |  |  |  | | | |  | | | | | | | |  |  |  |  | 50331648 | | | | |  | | | | | | | |  |  |  |  |  |  | |  | | | | | | | |
| адресуемые области указателей |  | | | | | | | |  | | | | | | | |  |  |  |  |  | | | |  | | | | | | | |  |  |  |  | p |  |  |  | |  | | | | | | | |  |  |  |  |  |  | |  | | | | | | | |
| значения в адресуемых областях |  | | | | | | | |  | | | | | | | |  |  |  |  |  | | | |  | | | | | | | |  |  |  |  | 0 |  |  |  |  | | | | | | | | |  |  |  |  |  |  |  | | | | | | | | |

x = &y;/\*присваиваем указателю x типа float\* адрес переменной y типа float\*/

printf("x=%p\t\*x=%f\ty=%f\n", x, \*x, y);

a =(int\*)x;/\*присваиваем указателю типа int\* указатель типа float\* с помощью явного приведения типов, a=x(000000000061FDF4),указатель а теперь указывает на переменную y \*/

\*a = \*x;/\*присваиваем значению типа int по адресу a(000000000061FDF4) значение типа float по адресу x(000000000061FDF4), числа типа int и float имееют различное представление, поэтому при выводе значения по адресу a дробная часть отбрасывается(программа выведет 3) \*/

pintf("a=%p\t\*a=%d\tx=%p\t\*x=%f\ty=%f\n", a, \*a, x, \*x, y);

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | 000000000061FDE0 | 000000000061FDE1 | 000000000061FDE2 | 000000000061FDE3 | 000000000061FDE4 | 000000000061FDE5 | 000000000061FDE6 | 000000000061FDE7 | | 000000000061FDE8 | 000000000061FDE9 | 000000000061FDEA | 000000000061FDEB | 000000000061FDEC | 000000000061FDED | 000000000061FDEE | 000000000061FDEF | | 000000000061FDF0 | | 000000000061FDF1 | | 000000000061FDF2 | | 000000000061FDF3 | | 000000000061FDF4 | 000000000061FDF5 | 000000000061FDF6 | 000000000061FDF7 | | 000000000061FDF8 | 000000000061FDF9 | 000000000061FDFA | 000000000061FDFB | 000000000061FDFC | 000000000061FDFD | 000000000061FDFE | 000000000061FDFF | | 000000000061FE00 | | 000000000061FE01 | | 000000000061FE02 | | 000000000061FE03 | | 000000000061FE04 | 000000000061FE05 | 000000000061FE06 | 000000000061FE07 | | 000000000061FE08 | 000000000061FE09 | 000000000061FE0A | 000000000061FE0B | 000000000061FE0C | 000000000061FE0D | 000000000061FE0E | 000000000061FE0F | 000000000061FE10 | 000000000061FE11 | 000000000061FE12 | 000000000061FE13 | (000000000061FE14) (000000000061FE15) (000000000061FE16) | 000000000061FE17 | | | 000000000061FE18 | 000000000061FE19 | 000000000061FE1A | 000000000061FE1B | 000000000061FE1C | 000000000061FE1D | 000000000061FE1E | 000000000061FE1F |
| переменные | n | | | | | | | | | m | | | | | | | | |  | |  | |  | |  | | y | | | | | x | | | | | | | | |  | |  | |  | |  | | b | | | | | a | | | | | | | | |  |  |  |  |  | c | | p | | | | | | | | |
| значения переменных |  | | | | | | | | |  | | | | | | | | |  | |  | |  | |  | | 3.5 | | | | | 000000000061FDF4 | | | | | | | | |  | |  | |  | |  | |  | | | | | 000000000061FDF4 | | | | | | | | |  |  |  |  |  |  | |  | | | | | | | | |
| адресуемые области указателей |  | | | | | | | | |  | | | | | | | | |  | |  | |  | |  | | x, a | | | | |  | | | | | | | | |  | |  | |  | |  | |  | | | | |  | | | | | | | | |  |  |  |  |  |  | |  | | | | | | | | |
| значения в адресуемых областях |  | | | | | | | | |  | | | | | | | | |  | |  | |  | |  | | 3 | | | | |  | | | | | | | | |  | |  | |  | |  | |  | | | | |  | | | | | | | | |  |  |  |  |  |  | |  | | | | | | | | |
| значения в адресуемых областях |  | | | | | | | | |  | | | | | | | | |  | |  | |  | |  | | 0.000000 | | | | |  | | | | | | | | |  | |  | |  | |  | |  | | | | |  | | | | | | | | |  |  |  |  |  |  |  | | | | | | | | | |

*p =(char\*)x;/\*указателю p типа char\* присваиваем указатель типа float\* \*/*

*c=\*p;/\*присваиваем переменной с типа char младший разряд переменной y (10110111)-4 байт(так как обратный порядок следования байт), так как разные типы char и float, получаем что переменная с=-73(число 10110111 отрицательное, поэтому переводим в обратный код =11001000, затем переводим в дополнительный код 11001001=-73 (так как обратный порядок следования байтов)\*/*

*\*p=\*(p+3) ;/\*изменяем четвертый байт на первый байт в переменной y типа float, с помощью указателей типа char\*, то есть 10110111(\*p)=01000110(\*(p+3))\*/*

*\*(p+3)=c;/\*меняем местами первый и четвертый байт в переменой y типа float, с помощью указателя типа char\* и переменной типа char( 01000110=10110111), теперь y=10110111010000001110011001000110(обратный порядок следования байт), что при переводе y=-0,000011\*/*

*printf("p=%p\tc=%d\tx=%p\ty=%f\n", p, c, x, y);*

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | 000000000061FDE0 | 000000000061FDE1 | 000000000061FDE2 | 000000000061FDE3 | 000000000061FDE4 | 000000000061FDE5 | 000000000061FDE6 | 000000000061FDE7 | | 000000000061FDE8 | 000000000061FDE9 | 000000000061FDEA | 000000000061FDEB | 000000000061FDEC | 000000000061FDED | 000000000061FDEE | 000000000061FDEF | | 000000000061FDF0 | | 000000000061FDF1 | | 000000000061FDF2 | | 000000000061FDF3 | | 000000000061FDF4 | | 000000000061FDF5 | | 000000000061FDF6 | | 000000000061FDF7 | | 000000000061FDF8 | 000000000061FDF9 | 000000000061FDFA | 000000000061FDFB | 000000000061FDFC | 000000000061FDFD | 000000000061FDFE | 000000000061FDFF | | 000000000061FE00 | | 000000000061FE01 | | 000000000061FE02 | | 000000000061FE03 | | 000000000061FE04 | 000000000061FE05 | 000000000061FE06 | 000000000061FE07 | | 000000000061FE08 | 000000000061FE09 | | 000000000061FE0A | | 000000000061FE0B | 000000000061FE0C | | 000000000061FE0D | | 000000000061FE0E | 000000000061FE0F | | 000000000061FE10 | | 000000000061FE11 | | | 000000000061FE12 | | 000000000061FE13 | | (000000000061FE14) (000000000061FE15) (000000000061FE16) | | 000000000061FE17 | | | 000000000061FE18 | 000000000061FE19 | 000000000061FE1A | 000000000061FE1B | 000000000061FE1C | 000000000061FE1D | 000000000061FE1E | 000000000061FE1F |
| переменные | n | | | | | | | | | m | | | | | | | | |  | |  | |  | |  | | y | | | | | | | | x | | | | | | | | |  | |  | |  | |  | | b | | | | | a | | | | | | | | | | | | |  | | | |  | |  | |  |  | | c | | | p | | | | | | | | |
| значения переменных |  | | | | | | | | |  | | | | | | | | |  | |  | |  | |  | | -0,000011 | | | | | | | | 000000000061FDF4 | | | | | | | | |  | |  | |  | |  | |  | | | | |  | | | | | | | | | | | | |  | | | |  | |  | |  |  | | -73 | | | 00000000061FDF4 | | | | | | | | |
| адресуемые области указателей |  | | | | | | | | |  | | | | | | | | |  | |  | |  | |  | | x | | | | | | | |  | | | | | | | | |  | |  | |  | |  | |  | | | | |  | | | | | | | | | | | | |  | | | |  | |  | |  |  | |  | | |  | | | | | | | | |
| значения в адресуемых областях |  | | | | | | | | |  | | | | | | | | |  | |  | |  | |  | | -0,000011 | | | | | | | |  | | | | | | | | |  | |  | |  | |  | |  | | | | |  | | | | | | | | | | | | |  | | | |  | |  | |  |  | |  | | |  | | | | | | | | |
| адресуемые области указателей |  | | | | | | | | |  | | | | | | | | |  | |  | |  | |  | | p | |  | |  | |  | |  | | | | | | | | |  | |  | |  | |  | |  | | | | |  | | |  | |  | | |  | |  | | |  | |  | | |  | |  | |  | |  | | |  | | | | | | | | |
| значения в адресуемых областях |  | | | | | | | | |  | | | | | | | | |  | |  | |  | |  | | 51 | |  | |  | |  | |  | | | | | | | | |  | |  | |  | |  | |  | | | | |  | | |  | |  | | |  | |  | | |  | |  | | |  | |  | |  | |  | |  | | | | | | | | | |

printf("p=%p\tc=%d\tx=%p\ty=%f\n", p, c, x, y);

m = &n;/\* присваиваем указателю m типа double адрес переменной n типа double\*/

printf("m=%p\t\*m=%lf\tn=%lf\n", m, \*m, n);

n = 5.5;/\*присваиваем переменной n=5.5 \*/

printf("m=%p\t\*m=%lf\tn=%lf\n", m, \*m, n);

b = n = y = 1.7;/\*присваиваем переменной y=1.7 типа float, присваиваем переменной типа double переменную типа float, происходит неявное приведение типов, поэтому n=1.7, переменной b тип int присваивается значение переменной типа double, поэтому дробная часть отбрасывается и b=1, после неявного приведения типов\*/

printf("b=%d\ty=%f\tn=%lf\n", b, y, n);

printf("\*a=%d\t\*x=%f\t\*m=%lf\n", \*a, \*x, \*m);

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | 000000000061FDE0 | 000000000061FDE1 | 000000000061FDE2 | 000000000061FDE3 | 000000000061FDE4 | 000000000061FDE5 | 000000000061FDE6 | 000000000061FDE7 | | 000000000061FDE8 | 000000000061FDE9 | 000000000061FDEA | 000000000061FDEB | 000000000061FDEC | 000000000061FDED | 000000000061FDEE | 000000000061FDEF | | 000000000061FDF0 | | 000000000061FDF1 | | 000000000061FDF2 | | 000000000061FDF3 | | 000000000061FDF4 | 000000000061FDF5 | 000000000061FDF6 | 000000000061FDF7 | | 000000000061FDF8 | 000000000061FDF9 | 000000000061FDFA | 000000000061FDFB | 000000000061FDFC | 000000000061FDFD | 000000000061FDFE | 000000000061FDFF | | 000000000061FE00 | | 000000000061FE01 | | 000000000061FE02 | | 000000000061FE03 | | 000000000061FE04 | 000000000061FE05 | 000000000061FE06 | 000000000061FE07 | | 000000000061FE08 | 000000000061FE09 | 000000000061FE0A | 000000000061FE0B | 000000000061FE0C | 000000000061FE0D | 000000000061FE0E | 000000000061FE0F | | 000000000061FE10 | | 000000000061FE11 | | 000000000061FE12 | | 000000000061FE13 | | (000000000061FE14) (000000000061FE15) (000000000061FE16) | | 000000000061FE17 | | 000000000061FE18 | 000000000061FE19 | 000000000061FE1A | 000000000061FE1B | 000000000061FE1C | 000000000061FE1D | 000000000061FE1E | 000000000061FE1F |
| переменные | n | | | | | | | | | m | | | | | | | | |  | |  | |  | |  | | y | | | | | x | | | | | | | | |  | |  | |  | |  | | b | | | | | a | | | | | | | | |  | |  | |  | |  | |  | | c | | p | | | | | | | | |
| значения переменных | 1.700000 | | | | | | | | | 000000000061FDF0 | | | | | | | | |  | |  | |  | |  | | 1.700000 | | | | | 000000000061FDF4 | | | | | | | | |  | |  | |  | |  | | 1 | | | | | 000000000061FE04 | | | | | | | | |  | |  | |  | |  | |  | |  | |  | | | | | | | | |
| адресуемые области указателей | m | | | | | | | | |  | | | | | | | | |  | |  | |  | |  | | x | | | | |  | | | | | | | | |  | |  | |  | |  | | a | | | | |  | | | | | | | | |  | |  | |  | |  | |  | |  | |  | | | | | | | | |
| значения в адресуемых областях | 1.700000 | | | | | | | | |  | | | | | | | | |  | |  | |  | |  | | 1.700000 | | | | |  | | | | | | | | |  | |  | |  | |  | | 1.000000 | | | | |  | | | | | | | | |  | |  | |  | |  | |  | |  | |  | | | | | | | | |

m += 2;/\*указатель m типа double на переменную n увеличиваем на два, то есть 000000000061FDE0+2\*8=000000000061FDF0 \*/

printf("n=%lf\tn=%p\tm=%p\n", n, &n, m);

\*m = (float)\*a - n + (int)\*x;/\*присваиваем значению по адресу m значение выражения (явное приведение типа значения по указателю a к типу float - переменная n тип double + явно приведённое типу int значение по адресу x) после неявногоо приведения типов к double мы получаем результат m=0.300000 так, как (1(a)-1,7+1=0,300000)

число y изменилось из-за того, что мы изменили число m, которая располагается в ячейках от 000000000061FDF0 до 000000000061FDF7, а число y располагается в ячейках памяти от 000000000061FDF4 до 000000000061FDF7; 0011111111010011001100110011001100110011001100110011001100110011-число m

Которое поменяло значение переменной y, которое стало равно первым четырём разрядам числа m y=00111111110100110011001100110011, из-за того, что под m типа double выделяется 8 байт, а под y типа float выделяется 4 байта, и обратного следования байт\*/

printf("m=%p\t\*m=%lf\n", m, \*m);

printf("…………);

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 000000000061FDE0 | 000000000061FDE1 | 000000000061FDE2 | 000000000061FDE3 | 000000000061FDE4 | 000000000061FDE5 | 000000000061FDE6 | 000000000061FDE7 | 000000000061FDE8 | 000000000061FDE9 | 000000000061FDEA | 000000000061FDEB | 000000000061FDEC | 000000000061FDED | 000000000061FDEE | 000000000061FDEF | 000000000061FDF0 | 000000000061FDF1 | 000000000061FDF2 | 000000000061FDF3 | 000000000061FDF4 | 000000000061FDF5 | 000000000061FDF6 | 000000000061FDF7 | 000000000061FDF8 | 000000000061FDF9 | 000000000061FDFA | 000000000061FDFB | 000000000061FDFC | 000000000061FDFD | 000000000061FDFE | 000000000061FDFF | 000000000061FE00 | 000000000061FE01 | 000000000061FE02 | 000000000061FE03 | 000000000061FE04 | 000000000061FE05 | 000000000061FE06 | 000000000061FE07 | 000000000061FE08 | 000000000061FE09 | 000000000061FE0A | 000000000061FE0B | 000000000061FE0C | 000000000061FE0D | 000000000061FE0E | 000000000061FE0F | 000000000061FE10 | 000000000061FE11 | 000000000061FE12 | 000000000061FE13 | (000000000061FE14) (000000000061FE15) (000000000061FE16) | 000000000061FE17 | 000000000061FE18 | 000000000061FE19 | 000000000061FE1A | 000000000061FE1B | 000000000061FE1C | 000000000061FE1D | 000000000061FE1E | 000000000061FE1F |
| переменные | n | | | | | | | | m | | | | | | | |  |  |  |  | y | | | | x | | | | | | | |  |  |  |  | b | | | | a | | | | | | | |  |  |  |  |  | c | p | | | | | | | |
| значения переменных | 1.700000 | | | | | | | | 000000000061FDF0 | | | | | | | |  |  |  |  | 1.650000 | | | | 000000000061FDF4 | | | | | | | |  |  |  |  | 1 | | | | 000000000061FE04 | | | | | | | |  |  |  |  |  |  |  | | | | | | | |
| адресуемые области указателей |  | | | | | | | |  | | | | | | | | m | | | | | | | |  | | | | | | | |  |  |  |  |  | | | |  | | | | | | | |  |  |  |  |  |  |  | | | | | | | |
| значения в адресуемых областях |  | | | | | | | |  | | | | | | | | 0.300000 | | | | | | | |  | | | | | | | |  |  |  |  |  | | | |  | | | | | | | |  |  |  |  |  |  |  | | | | | | | |
| адресуемые области указателей |  | | | | | | | |  | | | | | | | |  |  |  |  | x | | | |  | | | | | | | |  |  |  |  | a | | | |  | | | | | | | |  |  |  |  |  |  |  | | | | | | | |
| значения в адресуемых областях |  | | | | | | | |  | | | | | | | |  |  |  |  | 1.650000 | | | |  | | | | | | | |  |  |  |  | 1.000000 | | | |  | | | | | | | |  |  |  |  |  |  |  | | | | | | | |

После замены инструкции «m+=2;» инструкцией «m++;» результат работы программы такой:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | 000000000061FDE0 | 000000000061FDE1 | 000000000061FDE2 | 000000000061FDE3 | 000000000061FDE4 | 000000000061FDE5 | 000000000061FDE6 | 000000000061FDE7 | | 000000000061FDE8 | 000000000061FDE9 | 000000000061FDEA | 000000000061FDEB | 000000000061FDEC | 000000000061FDED | 000000000061FDEE | 000000000061FDEF | | 000000000061FDF0 | | 000000000061FDF1 | | 000000000061FDF2 | | 000000000061FDF3 | | 000000000061FDF4 | 000000000061FDF5 | 000000000061FDF6 | 000000000061FDF7 | | 000000000061FDF8 | 000000000061FDF9 | 000000000061FDFA | 000000000061FDFB | 000000000061FDFC | 000000000061FDFD | 000000000061FDFE | 000000000061FDFF | | 000000000061FE00 | | 000000000061FE01 | | 000000000061FE02 | | 000000000061FE03 | | 000000000061FE04 | 000000000061FE05 | 000000000061FE06 | 000000000061FE07 | | 000000000061FE08 | 000000000061FE09 | 000000000061FE0A | 000000000061FE0B | 000000000061FE0C | 000000000061FE0D | 000000000061FE0E | 000000000061FE0F | | 000000000061FE10 | | 000000000061FE11 | | 000000000061FE12 | | 000000000061FE13 | | (000000000061FE14) (000000000061FE15) (000000000061FE16)  3 | | 000000000061FE17 | | 000000000061FE18 | 000000000061FE19 | 000000000061FE1A | 000000000061FE1B | 000000000061FE1C | 000000000061FE1D | 000000000061FE1E | 000000000061FE1F |
| переменные | n | | | | | | | | | m | | | | | | | | |  | |  | |  | |  | | y | | | | | x | | | | | | | | |  | |  | |  | |  | | b | | | | | a | | | | | | | | |  | |  | |  | |  | |  | | c | | p | | | | | | | | |
| значения переменных | 1.700000 | | | | | | | | | 000000000061FDE8 | | | | | | | | |  | |  | |  | |  | | 1.700000 | | | | | 000000000061FDF4 | | | | | | | | |  | |  | |  | |  | | 1 | | | | | 000000000061FE04 | | | | | | | | |  | |  | |  | |  | |  | |  | |  | | | | | | | | |
| адресуемые области указателей |  | | | | | | | | | m | | | | | | | | |  | |  | |  | |  | | x | | | | |  | | | | | | | | |  | |  | |  | |  | | a | | | | |  | | | | | | | | |  | |  | |  | |  | |  | |  | |  | | | | | | | | |
| значения в адресуемых областях |  | | | | | | | | |  | | | | | | | | |  | |  | |  | |  | | 1.700000 | | | | |  | | | | | | | | |  | |  | |  | |  | | 1.000000 | | | | |  | | | | | | | | |  | |  | |  | |  | |  | |  | |  | | | | | | | | |

После прибавления 1 к указателю m на переменную n, то m теперь указывает сам на себя, поэтому при попытке вытащить значение по адресу m, то мы вытащим адрес указателя m и мы не можем изменить его, так как не сможем присвоить указателю с адресом на себя записать выражение, так как указатель содержит адрес на ячейку в памяти, происходит ошибка.

**Задание 3.**

Объявить по две переменные типов *char*, *int* и *double*, а также указатель на *char.* Вывести на экран размеры и адреса всех переменных, начертить схему расположения переменных в памяти. Поменять порядок объявления переменных, запустить программу повторно, проанализировать, что изменилось. Задать переменной типа *int* такое значение, чтобы значение каждого байта было уникальным, использовать для этого шестнадцатеричную константу. Записать адрес этой переменной в указатель на *char* и с его помощью вывести на экран содержимое каждого байта (тоже в шестнадцатеричной системе счисления). Проанализировать, прямой или обратный порядок расположения байт при записи числа применяется в используемой системе.

Повторить выполнение этого задания на другой платформе. Сравнить результаты работы программы на разных платформах, сделать выводы.

1. Операционная система Kubuntu 18.04.2, IDE Code::Blocks

Текст программы (первый порядок объявления переменных):

#include <stdio.h>

#include <stdlib.h>

int main()

{

char x, y;

int a, b=0xFA5E4B9C;

double n, m;

char \*uc=&b;

printf("char:\tadress x=%p, sizeof(x)=%d\tadress y=%p, sizeof(y)=%d\n",&x, sizeof(x), &y, sizeof(y));

printf("int:\tadress a=%p, sizeof(a)=%d\tadress b=%p, sizeof(b)=%d\n",&a, sizeof(a), &b, sizeof(b));

printf("double:\tadress n=%p, sizeof(n)=%d\tadress m=%p, sizeof(m)=%d\n",&n, sizeof(n), &m, sizeof(m));

printf("char pointer uc adress=%p, sizeof(uc)=%d\n", &uc, sizeof(uc));

for (x=0; x<4;x++)

{

printf("%d bite = %hhx\n",x+1, \*uc);

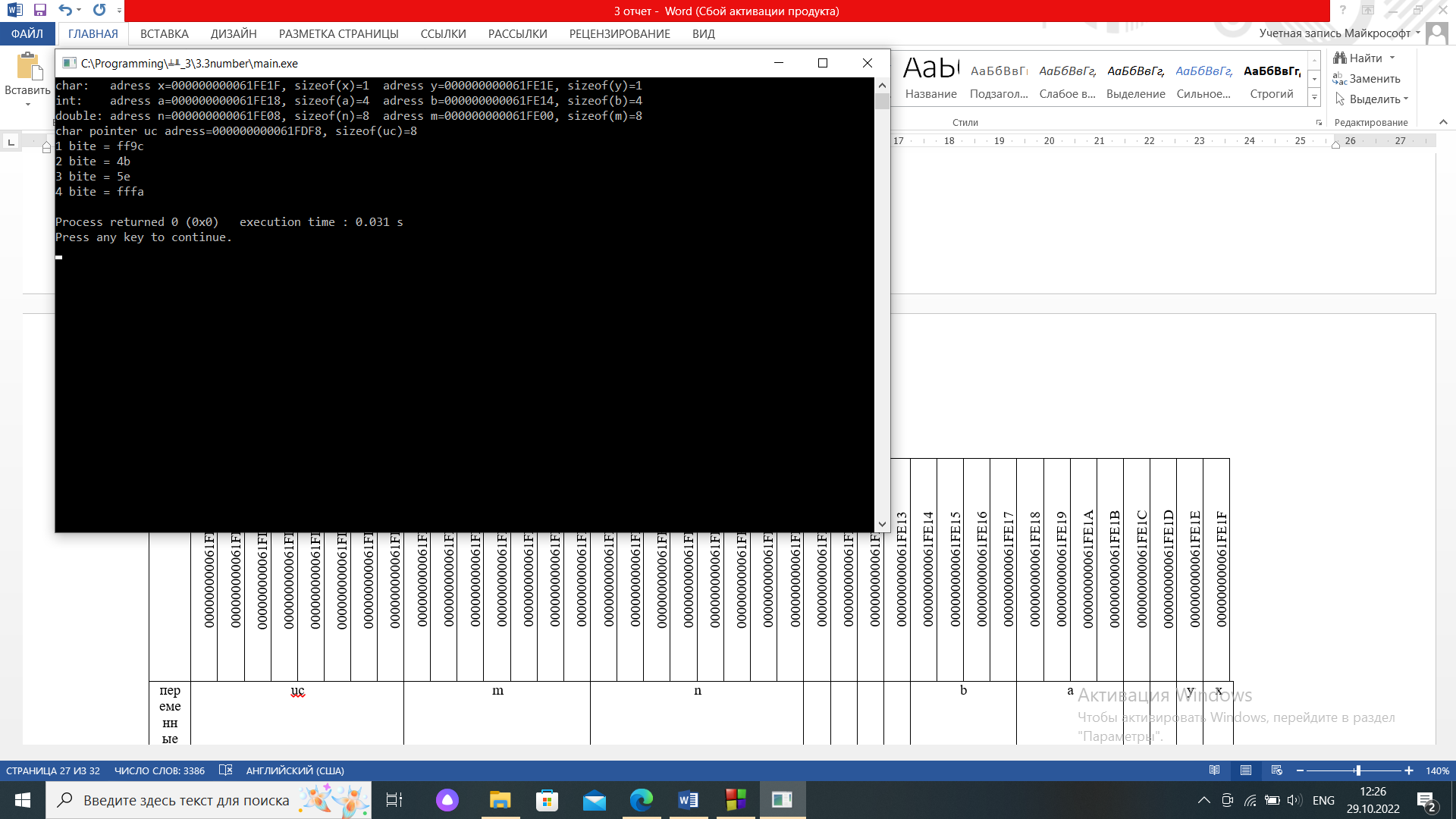
uc++;

}

return 0;

}

Результаты работы программы:



Размещение переменных в памяти

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| адреса | 000000000061FDF8 | 000000000061FDF9 | 000000000061FDFA | 000000000061FDFB | 000000000061FDFC | 000000000061FDFD | 000000000061FDFE | 000000000061FDFF | 000000000061FE00 | 000000000061FE01 | 000000000061FE02 | 000000000061FE03 | 000000000061FE05 | 000000000061FE06 | 000000000061FE07 | 000000000061FE08 | 000000000061FE09 | 000000000061FE0A | 000000000061FE0B | 000000000061FE0C | 000000000061FE0D | 000000000061FE0E | 000000000061FE0F | 000000000061FE10 | 000000000061FE11 | 000000000061FE12 | 000000000061FE13 | 000000000061FE14 | 000000000061FE15 | 000000000061FE16 | 000000000061FE17 | 000000000061FE18 | 000000000061FE19 | 000000000061FE1A | 000000000061FE1B | 000000000061FE1C | 000000000061FE1D | 000000000061FE1E | 000000000061FE1F |
| переменные | uc | | | | | | | | m | | | | | | | n | | | | | | | |  |  |  |  | b | | | | a | | | |  |  | y | x | |

Текст программы (второй порядок объявления переменных):

#include <stdio.h>

#include <stdlib.h>

int main()

{

char x;

int a;

char y;

double m, n;

int b=0xFA5E4B9C;

char \*uc=&b;

printf("char:\tadress x=%p, sizeof(x)=%d\tadress y=%p, sizeof(y)=%d\n",&x, sizeof(x), &y, sizeof(y));

printf("int:\tadress a=%p, sizeof(a)=%d\tadress b=%p, sizeof(b)=%d\n",&a, sizeof(a), &b, sizeof(b));

printf("double:\tadress n=%p, sizeof(n)=%d\tadress m=%p, sizeof(m)=%d\n",&n, sizeof(n), &m, sizeof(m));

printf("char pointer uc adress=%p, sizeof(uc)=%d\n", &uc, sizeof(uc));

for (x=0; x<4;x++)

{

printf("%d bite = %hhx\n",x+1, \*uc);

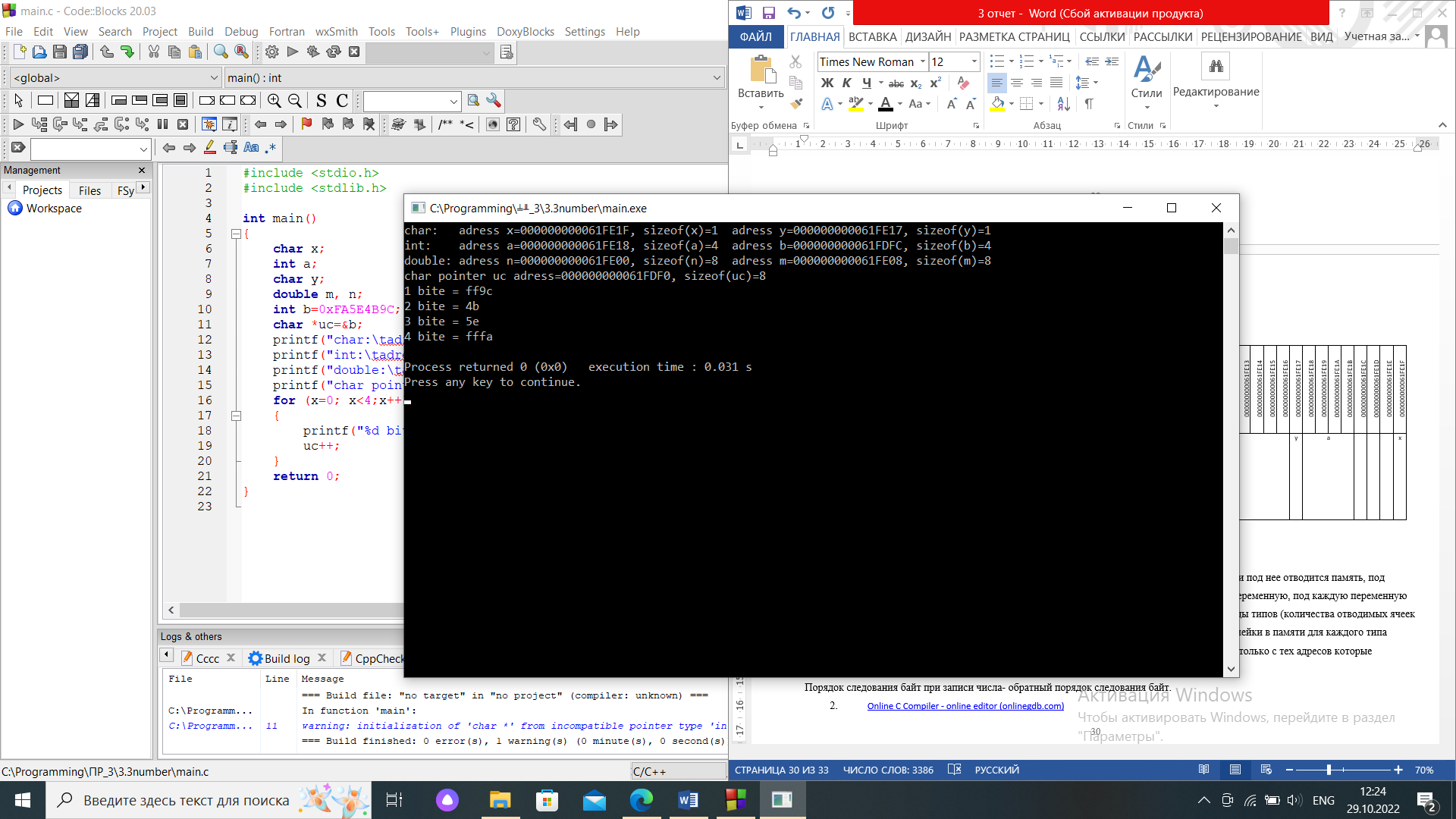
uc++;

}

return 0;

}

Результаты работы программы:



Размещение переменных в памяти

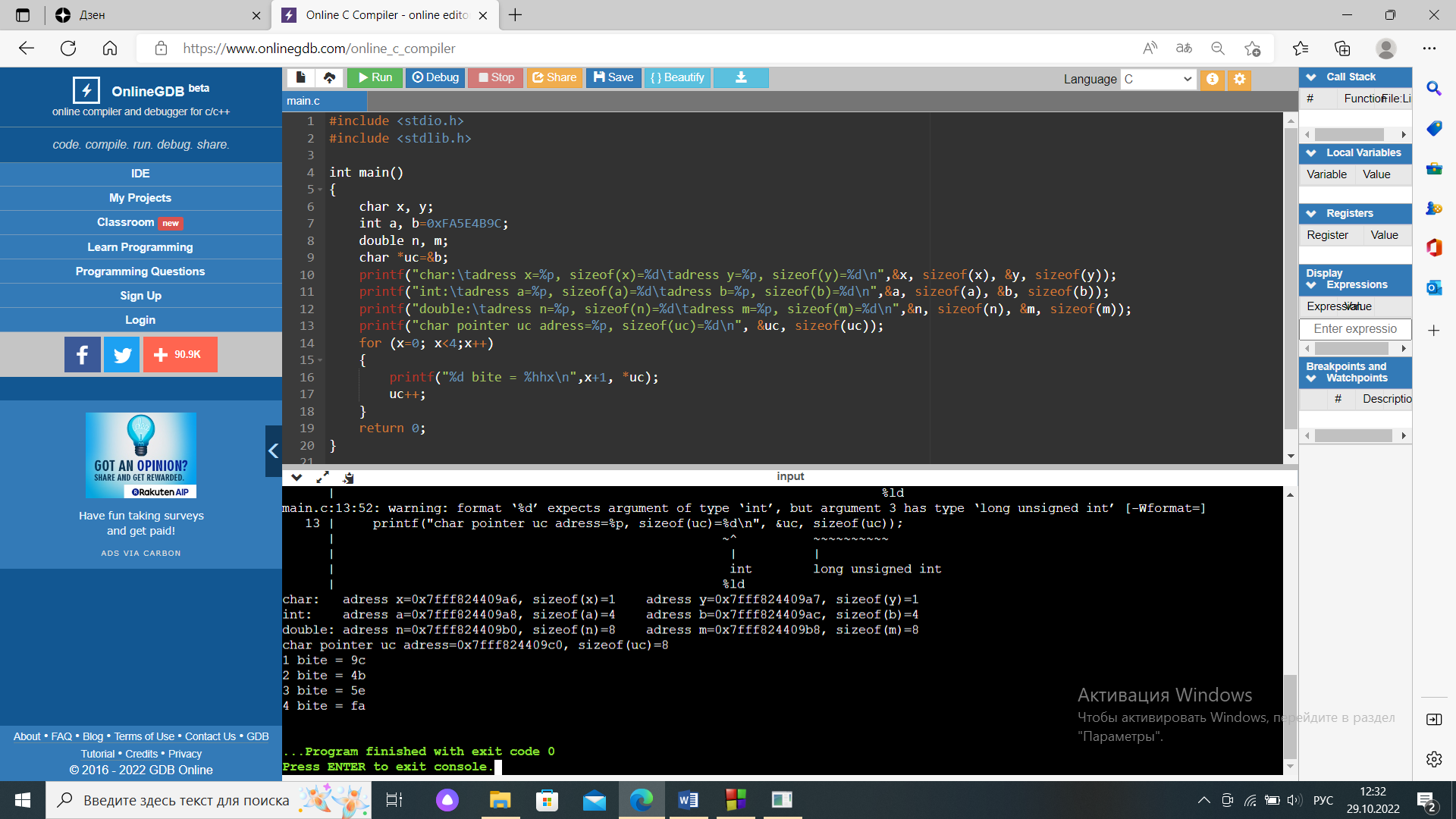
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| адреса | 000000000061FDF0 | 000000000061FDF1 | 000000000061FDF2 | 000000000061FDF3 | 000000000061FDF4 | 000000000061FDF5 | 000000000061FDF6 | 000000000061FDF7 | 000000000061FDF8 | 000000000061FDF9 | 000000000061FDFA | 000000000061FDFB | 000000000061FDFC | 000000000061FDFD | 000000000061FDFE | 000000000061FDFF | 000000000061FE00 | 000000000061FE01 | 000000000061FE02 | 000000000061FE03 | 000000000061FE04 | 000000000061FE05 | 000000000061FE06 | 000000000061FE07 | 000000000061FE08 | 000000000061FE09 | 000000000061FE0A | 000000000061FE0B | 000000000061FE0C | 000000000061FE0D | 000000000061FE0E | 000000000061FE0F | 000000000061FE10 | 000000000061FE12 | 000000000061FE13 | 000000000061FE14 | 000000000061FE15 | 000000000061FE16 | 000000000061FE17 | 000000000061FE18 | 000000000061FE19 | 000000000061FE1A | 000000000061FE1B | 000000000061FE1C | 000000000061FE1D | 000000000061FE1E | 000000000061FE1F |
| переменные | uc | | | | | | | |  |  |  |  | b | | | | n | | | | | | | | m | | | | | | | |  | | | | | | y | a | | | |  |  |  | x |

*Выводы:* Память в операционной системе на windows работает так: объявляется переменная и под нее выделяется память, следующая объявленная переменная, располагается в памяти до предыдущей переменной, под каждую переменную выделяется столько ячеек памяти сколько байт в памяти отводится под тип переменной. Память в операционной системе выделяется в соответствие с объявлением переменных, размещение не зависит от объёма памяти выделяемого на переменную, не зависит от типов переменных.

Порядок следования байт при записи числа- обратный порядок следования байт.

1. [Online C Compiler - online editor (onlinegdb.com)](https://www.onlinegdb.com/online_c_compiler)

Результаты работы программы при первом порядке объявления переменных:



|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| адреса | 0x7fff824409a6 | 0x7fff824409a7 | 0x7fff824409a8 | 0x7fff824409a9 | 0x7fff824409aa | 0x7fff824409ab | 0x7fff824409ac | 0x7fff824409ad | 0x7fff824409ae | 0x7fff824409af | 0x7fff824409b0 | 0x7fff824409b1 | 0x7fff824409b2 | 0x7fff824409b3 | 0x7fff824409b4 | 0x7fff824409b5 | 0x7fff824409b6 | 0x7fff824409b7 | 0x7fff824409b8 | 0x7fff824409b9 | 0x7fff824409ba | 0x7fff824409bb | 0x7fff824409bc | 0x7fff824409bd | 0x7fff824409be | 0x7fff824409bf | 0x7fff824409c0 | 0x7fff824409c1 | 0x7fff824409c2 | 0x7fff824409c3 | 0x7fff824409c4 | 0x7fff824409c5 | 0x7fff824409c6 | 0x7fff824409c7 |
| переменные | x | y | a | | | | b | | | | n | | | | | | | | m | | | | | | | | uc | | | | | | | |

Размещение переменных в памяти

Результаты работы программы при втором порядке объявления переменных:

#include <stdio.h>

#include <stdlib.h>

int main()

{

char y;

int a;

char x;

double n, m;

int b=0xFA5E4B9C;

char \*uc=&b;

printf("char:\tadress x=%p, sizeof(x)=%d\tadress y=%p, sizeof(y)=%d\n",&x, sizeof(x), &y, sizeof(y));

printf("int:\tadress a=%p, sizeof(a)=%d\tadress b=%p, sizeof(b)=%d\n",&a, sizeof(a), &b, sizeof(b));

printf("double:\tadress n=%p, sizeof(n)=%d\tadress m=%p, sizeof(m)=%d\n",&n, sizeof(n), &m, sizeof(m));

printf("char pointer uc adress=%p, sizeof(uc)=%d\n", &uc, sizeof(uc));

for (x=0; x<4;x++)

{

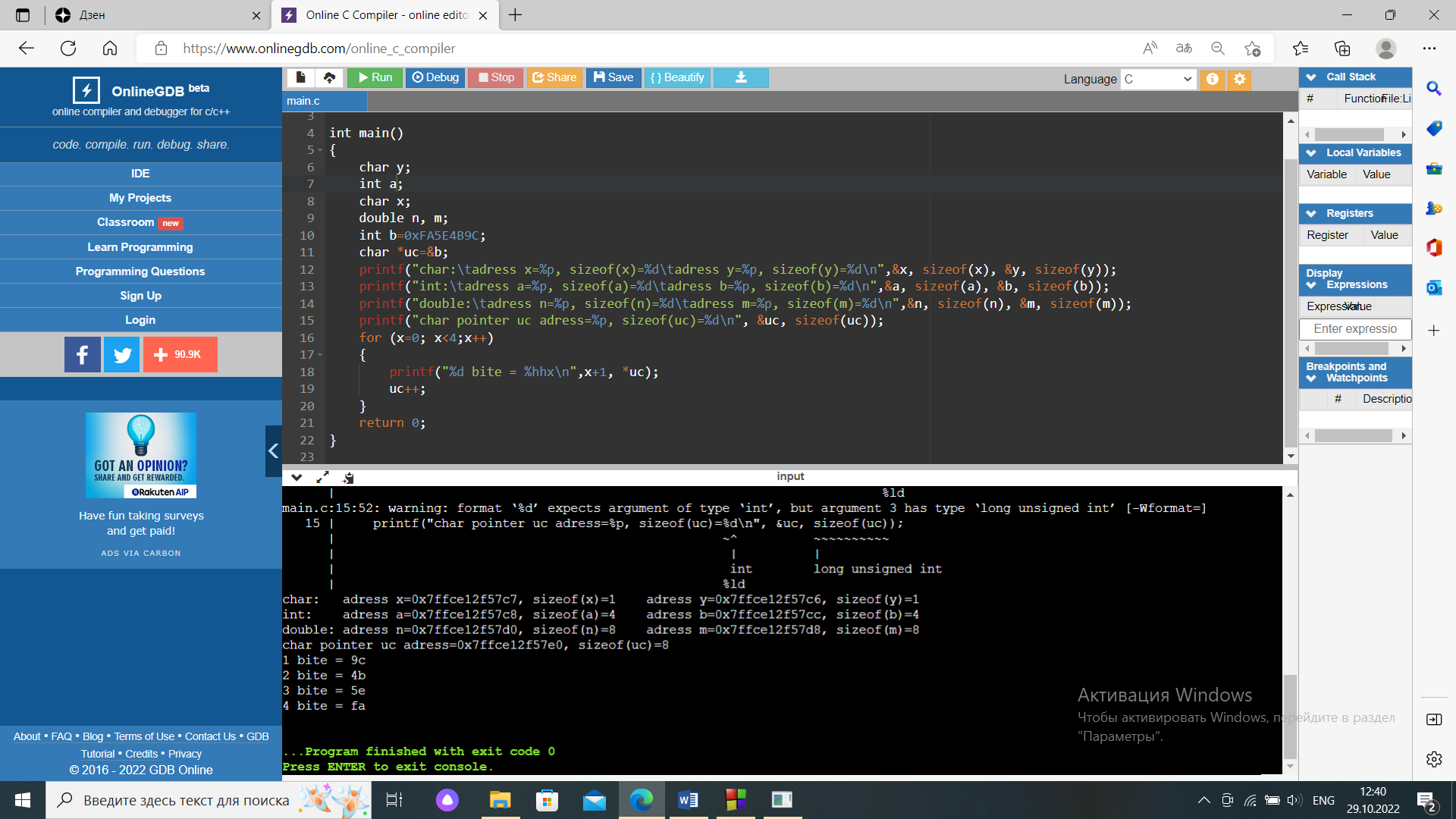
printf("%d bite = %hhx\n",x+1, \*uc);

uc++;

}

return 0;

}



Размещение переменных в памяти

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| адреса | 0x7ffce12f57c6 | 0x7ffce12f57c7 | 0x7ffce12f57c8 | 0x7ffce12f57c9 | 0x7ffce12f57ca | 0x7ffce12f57cb | 0x7ffce12f57cc | 0x7ffce12f57cd | 0x7ffce12f57ce | 0x7ffce12f57cf | 0x7ffce12f57d0 | 0x7ffce12f57d1 | 0x7ffce12f57d2 | 0x7ffce12f57d3 | 0x7ffce12f57d4 | 0x7ffce12f57d5 | 0x7ffce12f57d6 | 0x7ffce12f57d7 | 0x7ffce12f57d8 | 0x7ffce12f57d9 | 0x7ffce12f57da | 0x7ffce12f57db | 0x7ffce12f57dc | 0x7ffce12f57dd | 0x7ffce12f57de | 0x7ffce12f57df | 0x7ffce12f57e0 | 0x7ffce12f57e1 | 0x7ffce12f57e2 | 0x7ffce12f57e3 | 0x7ffce12f57e4 | 0x7ffce12f57e5 | 0x7ffce12f57e6 | 0x7ffce12f57e7 |
| переменные | y | x | a | | | | b | | | | n | | | | | | | | m | | | | | | | | uc | | | | | | | |

*Выводы:* Память в операционной системе на linux работает так: наименьший адрес в памяти получает та переменная, под которую отводится меньшее количество байт, если таких несколько, то наименьший адрес получает та переменная, которая объявлена раньше, следующими идут переменные, под которые отводится большее количество байт. В операционной системе на Linux, память выделяется в зависимости от типа переменной (то есть от размера выделяемой памяти под тип переменной).

Порядок следования байт при записи числаобратный порядок следования байт.

*Выводы:* Главное различие выделения памяти в операционной системе на windows от linux, в том, что на linux выделение памяти зависит от объема памяти необходимого для хранения переменной определенного типа, на windows же выделение зависит от порядка объявления переменных и не зависит от объема памяти необходимого для хранения переменной (объявляется первая переменная, занимает место в памяти, и все следующие переменные размещаются в памяти до первой объявленной переменной).