Элементы ЯП C/C++

Алгоритмы
и алгоритмические языки

goo.gl/c8pyqx

Лекция 4, 26 сентября, 2017

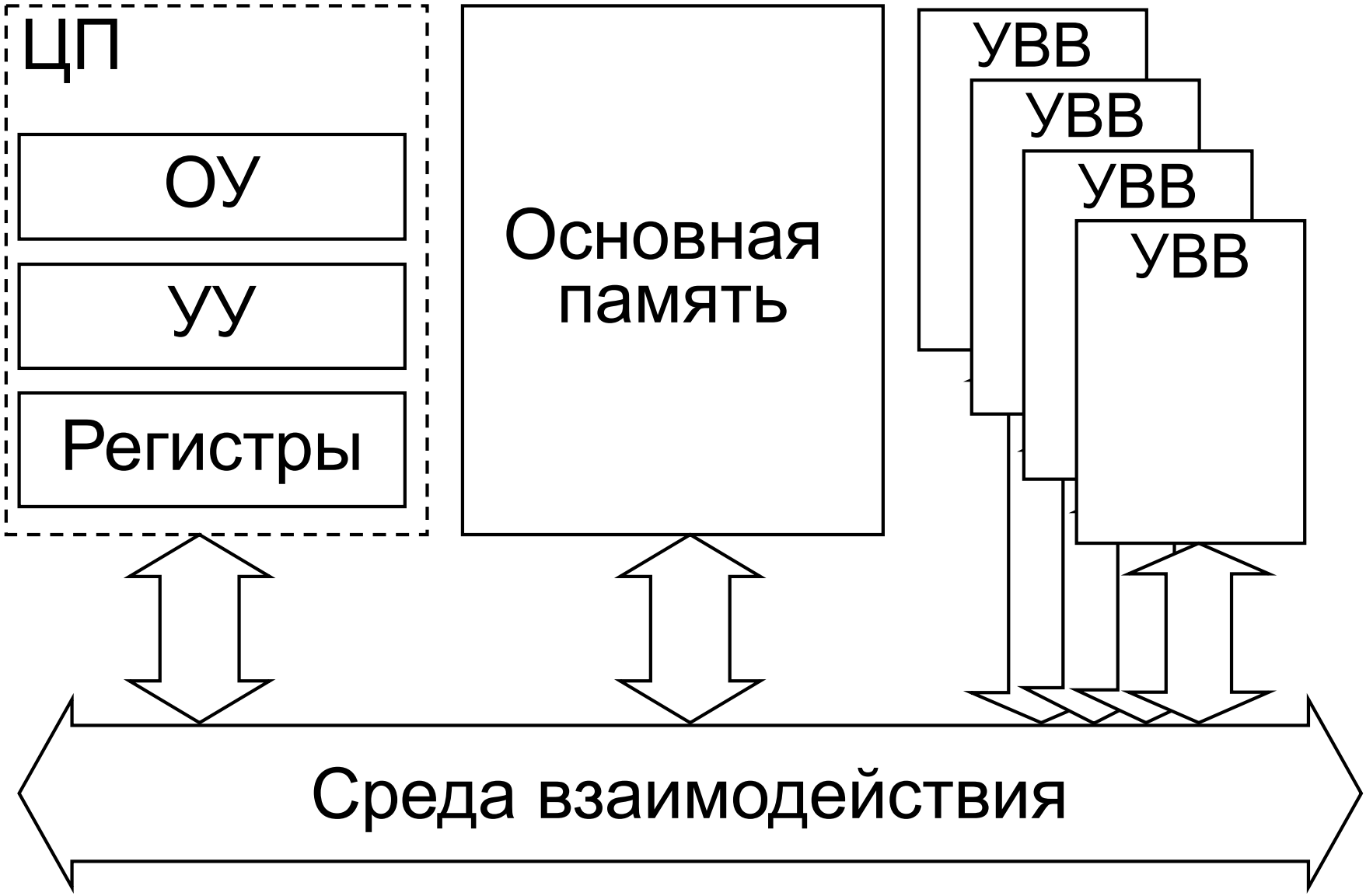
Лектор:

Дмитрий Северов, кафедра информатики 608 КПМ

dseverov@mail.mipt.ru

http://cs.mipt.ru/wp/?page_id=6077

Модель исполнителя



Переменная

Именованная область памяти типизованных значений.

1. **Имя (адрес начала),**
2. **Тип (размер и правила операций),**
3. **Значение (содержимое памяти)**

Эволюция базовых типов данных

- **Базовые типы данных**
  - Целые (char, int)
  - Вещественные (float, double)
  - Никакой (void)
- **Модификаторы типа**
  - Без/Знаковые (un/signed)
  - Короткие/Длинные (short, long)
- **Спецификаторы класса памяти:**
  - Статические/Автоматические (static, auto)
  - Регистровые, Внешние (register, extern)
- **Квалификаторы изменчивости:**
  - Неизменяемые (const),
  - Изменчивые (volatile), исключаемые из оптимизации

Приоритет операций

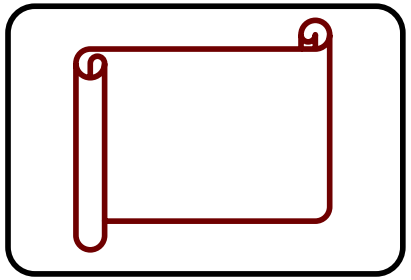
Приоритет и порядок	Обозначение	Назначение
1⇨	::	Разрешения контекста
2⇦	()	Вызов или описание функции
	()	Конструкция значения
	[]	Индекс массива
	->	Косвенная принадлежность
	.	Прямая принадлежность
	++	Инкремент (постфиксная)
	--	Декремент (постфиксная)
3⇨	!	Логическое отрицание
	~	Битовое отрицание (сумма по модулю 2)
	+	Унарный плюс
	-	Унарный минус
	++	Инкремент (префиксная)

Приоритет и порядок	Операция	Назначение
3⇨	--	Декремент (префиксная)
	&	Адрес
	*	Разыменование
	()	Преобразование типа
	sizeof	Размер в байтах
	new	Динамическое выделение памяти
	delete	Динамическое освобождение памяти
4⇨	.*	Разыменование члена
	->*	Косвенное разыменование члена
5⇨	*	Умножение
	/	Деление
	%	Остаток от деления
6⇨	+	Сложение
	-	Вычитание
7⇨	<<	Сдвиг влево
	>>	Сдвиг вправо

Приоритет и порядок	Операция	Назначение
8⇨	<	Меньше
	<=	Меньше или равно
	>=	Больше или равно
	>	Больше
9⇨	==	Равно
	!=	Не равно
10⇨	&	Побитовая конъюнкция
11⇨	^	Побитовое сложение по модулю два
12⇨	|	Побитовая дизъюнкция
13⇨	&&	Конъюнкция
14⇨	||	Дизъюнкция
15⇨	?:	Условное выражение

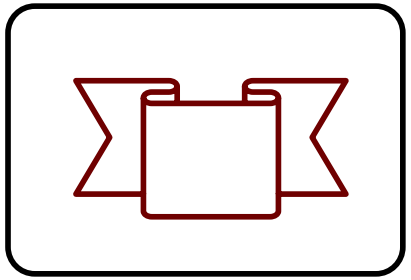
Приоритет и порядок	Операция	Назначение
16⇨	=	Присваивание
	*=	Умножение и присваивание
	/=	Деление и присваивание
	%=	Нахождение остатка и присваивание
	+=	Сложение и присваивание
	-=	Вычитание и присваивание
17⇨	&=	Побитовая конъюнкция и присваивание
	^=	Побитовое сложение по модулю два и присваивание
	|=	Побитовая дизъюнкция и присваивание
	<<=	Сдвиг влево и присваивание
	>>=	Сдвиг вправо и присваивание
18⇨	throw	Генерация исключения
19⇨	,	Объединение двух выражений в одно

Составные типы данных



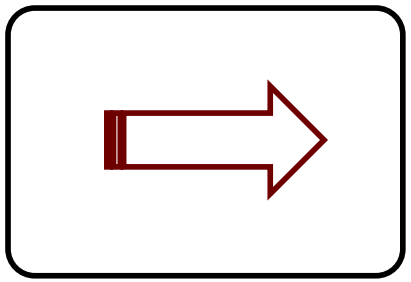
Массивы – наборы нумерованных однородных данных

- Строки



Структуры – наборы именованных разнородных данных

- Объединения
- Перечисления



Указатели – переменные содержащие адрес

- Ссылки

Массивы

- **Описание**

```
int a[3],b[2][3];
```

- **Описание с инициализацией**

```
char c[] = {'0','2','4'};
int d[][3] =
 {{1,2,3},{4,5},{6}};
```

- **Доступ к элементу**

```
a[0] = d[0][1] + 1;
```

Строки

- **Одномерные массивы однобайтных целых элементов (символов), содержащие последним элемент со значением ноль.**

- ```
  char Str1[] = {'H','e','l','l','o','\0'};
  char Str2[5] = {',',' '};
  char Str3[] = "world";
  ```

- **Использование**

  ```
  cout << Str1 << Str2 << Str3 << endl;
  ```
  ```

Структуры

- **Определение конструкции набора данных:**

```
struct My_type {
        type_a Member_1,Member_2;
        type_b Member_3;… } My_object;
```

- **Частные случаи:**

```
struct Complex { double Re,Im; };
struct Complex A;
struct  { double Re,Im; } B;
```

- **Доступ к полям:** `B.Re` **или** `A.Im`

Объединение

совмещение полей — точек зрения

union *тег* { *список_полей* } *имя_объекта*;

Перечисление

Набор именованных целых констант

enum *тег* { *список_имен* } *имя_объекта*;

Битовые поля

Набор целых полей определённой длины

struct *тег* { *поле*:*число_бит*; } *имя*;

union *тег* { *поле*:*число_бит*; } *имя*;

A								
B	B[0]=Y		B[1]=Z		B[2]=X		B[3]=Z	
X	a	b	c	d	e	f	g	h
W	L_word				H_word			

```
#include <iostream>
using namespace std;

enum {Z=0,X,Y=4};
union {
    int A;
    char B[4];
    struct { int a:4,b:4,c:4,d:4,e:4,f:4,g:4,h:4;} X;
    struct { short int L_word, H_word;} W;
} N;

int main() {
    N.B[0]=Y; N.B[1]=Z; N.B[2]=X; N.B[3]=Z;
    cout << N.W.H_word << " " << N.W.L_word << endl;
    cout << N.A << endl;
    cout << N.X.h << N.X.g << N.X.f << N.X.e
         << N.X.d << N.X.c << N.X.b << N.X.a << endl;
    return 0;
}
```

```
1 4
65540
00010004
```

УКАЗАТЕЛИ

- **Переменные, содержащие адрес**

- **Описание и инициализация**

```
int *Pc, **Pd;
int a=5, b[]={1,2,3}, *Pa=&a, *Pb=b;
char *s="String";
```

- **Присваивание значения**

```
const int *b =
{1,2,3};
```

```
Pd=&Pb; Pc=&b[1];
```

Операции с указателями

```
int n, b, a[10], *pb=&a[0], *pe=&a[9], *pc;
```

- **Арифметические**

```
n = pe - pb;   // кол-во элементов (не байт!)
pb++;          // &a[1] (адрес следующего
               // элемента но не байта!)
pc = pe – 1;   // &a[8]
pb + pe;       // ОШИБКА !!!
```

- **Разыменования**

```
b = *pc;       // b = a[8];
```

NB: операции с указателями являются размерными!!!

Операции с указателями

```
struct complex { double Re; double Im; } a, *p;
p = &a;
```

- **Разыменования**

```
a.Re = 1.;
p->Im = 0.;
```

NB: операции с указателями являются размерными!!!

Универсальные указатели

- `void *P, *Q;`
- **Запрещены операции**
  - Арифметические
  - Разыменования
- **Типичное применение**

```
P=malloc(sizeof(double));  // P = new double;
Q=calloc(N,sizeof(int));   // Q = new(int[N]);
free(Q);                   // delete(Q);
Q=realloc(Q,M);
```

- **Универсальный указатель можно**
  - Инициализировать значением указателя любого типа
  - Преобразовать в указатель любого типа

```
#include <iostream>
using namespace std;

int main() {
   const int n=64;
   char  str[] = "Hello, World!\n";
   void  *p, *ps=str;
   char **pps=(char**)&ps;
   int *a, *pb, *pe;

   cout << str << ps << " " << pps << ' ' << (**pps) << endl;
   p=malloc(4*n); pb=a=(int*)p; for (int i=0;i<n;i++) a[i]=i;
   cout<<pb<<"=pb\t  pb+1="<<pb+1<<"\n";
   cout<<pb+2<<": a[2]= "<<*(pb+2)<< endl;
   free(a); pb=a=NULL;
   cout << pb << endl;

   pb=a=new int[n]; for(int i=0;i<n;i++) *(a+i)=10*i;
   cout << a << "\t"  <<  *(pb+2) <<endl;
   pe=&a[n-1];
   cout <<pe-pb<< endl;
   a=(int*)realloc(a,2*n);
   cout<<"a="<< a <<"\ta[0]="<<*a<<
    "\na[n-1]="<<a[n-1]<<" a[n]="<<a[n]<<endl;
}
```

```
Hello, World!
0x7fff5fbff559 0x7fff5fbff4e0 H
0x100519920=pb pb+1=0x100519924
0x100519928: a[2]= 2
0x0
0x100519920 20
63
a=0x100519920 a[0]=0
a[n-1]=630 a[n]=64
```

Ссылки*)

- **Другое имя переменной (псевдоним)**
- **Описание и инициализация**

```
int a=5, &b=a;
                // a и b — разные имена одной
                // и той же области памяти
```

*) только в C++

Dev-C++ 4.9.9.2

Файл Правка Поиск Вид Проект Выполнить Отладка Сервис CVS Окно Справка

Создать Вставить Переключить Перейти

Проект | Классы | Отладка

tst1.cpp

```
#include "iostream"

using namespace std;

int main() {
int a=5, &b=a, c=10;

    cout << " b= " << b << " &b= " << &b << endl;
    b=c;
    cout << " b= " << b << " &b= " << &b << "\ta= " << a << endl;

    return 0;
}
```

D:\A1\Lect2013\Exmp\4\tst1.exe

```
b= 5 &b= 0x22ff44
b= 10 &b= 0x22ff44      a= 10
```

Компилятор | Ресурсы | Журнал компиляции | Отладка | Результаты поиска | Закрыть

Отладка | Отслеживать | Вывод

Следующий шаг | Продолжить выполнение | Отладка | Добавить в наблюдаемы

Шаг внутрь | Выполнить до курсора | Остановить выполнение | Удалить объект наблюдени

12: 9 | Вставка | 15 строк в файле

Тип FILE*

```
typedef struct _iobuf FILE;
```

Или

```
#define FILE struct _iobuf

FILE *a;
FILE* fopen(
        const char* name,
        const char* access
    );
int fclose(FILE* a);
```

```
struct _iobuf {
    char *_ptr;
    int _cnt;
    char *_base;
    int _flag;
    int _file;
    int _charbuf;
    int _bufsiz;
    char
*_tmpfname;  };
```

Определение новых действий

Функции

Соглашения функции

1. Имеет список аргументов (параметров)
2. Возвращает значение
3. Объявляется указанием своего имени и типа возвращаемого значения
4. Определяется перечислением операторов

D:\A1\Lect2015\Exmp\4\tst20.c - Dev-C++ 5.10

Файл Правка Поиск Вид Проект Выполнить Сервис AStyle Окно Справка

TDM-GCC 4.8.1 32-bit Debug

(globals)

Проект Класс

f(int n) : int
main() : int

tst20.c

```c
#include <stdio.h>

inline int f(int n) { return n?n*f(n-1):1; }

int main() {
int N;
    scanf("%d",&N);
    printf("%d! = %d\n",N,f(N));

    return 0;
}
```

D:\A1\Lect2015\Exmp\4\tst20.exe

```
7
7! = 5040

--------------------------------
Process exited after 3.1 seconds with return value 0
Для продолжения нажмите любую клавишу . . .
```

Компилятор Ресурсы Журнал компиляции Отладка Результаты по

Прервать

Shorten compiler paths

```
Compilation results...
--------
- Errors: 0
- Warnings: 0
- Output Filename: D:\A1\Lect2015\Exmp\4\tst2
- Output Size: 121,37890625 KiB
- Compilation Time: 1,01s
```

Line: 3 Col: 12 Sel: 0 Lines: 11 Length: 158 Вставка Done parsing in 0,015 seconds

EN 0:00

Пример 1: решето Эратосфена

Создание динамического массива

2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0

Заполнение признаком простоты

Dev-C++ 4.9.9.2

Файл Правка Поиск Вид Проект Выполнить Отладка Сервис CVS Окно Справка

Свернуть

Созда

tst3.cpp

```
#include <stdio.h>

int* R(int);

int main() {
int N, *a;
    scanf("%d",&N); a=R(N);
    for(int i=2; i<N; i++) if(a[i]) printf("%d ",i); printf("\n");
    delete(a);

    return 0;
}

int* R(int N) {
int *a = new int[N],k;
    for(int i=2; i<N; i++) a[i]=1;
    for(int i=2; i<N; i++) if(a[i]) for(int j=i;(k=i*j)<N;j++) a[k]=0;
    return a;
}
```

Компилятор | Ресурсы | Журнал компиляции | Отладка | Результаты поиска

Ready.

Dev-C++ 4.9.9.2

Файл Правка Поиск Вид Проект Выполнить Отладка Сервис CVS Окно Справка

Свернуть

Созда

tst3.cpp

```
#include <stdio.h>

int* R(int);

int main() {
int N, *a;
    scanf("%d",&N); a=R(N);
    for(int i=2; i<N; i++) if(a[i]) printf("%d ",i); printf("\n");
    delete(a);

    return 0;
}

int* R(int N)
int *a = new i
    for(int i=
    for(int i=2; i<N; i++) if(a[i]) for(int j=i;(k=i*j)<N;j++) a[k]=0;
    return a;
}
```

D:\A1\Lect2013\Exmp\4\tst3.exe

```
100
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97
```

D:\A1\Lect2013\Exmp\4\tst3.exe

```
1000
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97 101 103 1
07 109 113 127 131 137 139 149 151 157 163 167 173 179 181 191 193 197 199 211 2
23 227 229 233 239 241 251 257 263 269 271 277 281 283 293 307 311 313 317 331 3
37 347 349 353 359 367 373 379 383 389 397 401 409 419 421 431 433 439 443 449 4
57 461 463 467 479 487 491 499 503 509 521 523 541 547 557 563 569 571 577 587 5
93 599 601 607 613 617 619 631 641 643 647 653 659 661 673 677 683 691 701 709 7
19 727 733 739 743 751 757 761 769 773 787 797 809 811 821 823 827 829 839 853 8
57 859 863 877 881 883 887 907 911 919 929 937 941 947 953 967 971 977 983 991 9
97
```

Компилятор | Ресурсы | Журнал компиляции | Отладка | Результаты поиска

Ready.

Пример 1: решето Эратосфена

Создание динамического массива

2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0

Создание двумерного динамического массива

```
const int n=3, m=5;
int (*a)[m] = new int[n][m];   // *b[m]
                // a = (int(*)[m])malloc(n*m*4);
```

4 байта

20 байт

Dev-C++ 4.9.9.2

Файл Правка Поиск Вид Проект Выполнить Отладка Сервис CVS Окно Справка

Создать Вставить Переключить Перейти

Проект | Классы | Отладка

tst10.cpp

```
const int K=2, N=3, M=4;

int main() {
char (*a)[M] = new char[N][M], *pa = (char*)a;
     for(int n=0; n<N; n++)
              for(int m=0; m<M; m++) a[n][m] = 65+M*n+m;
     for(int i=0; i<N*M; i++) cout << *(pa+i) << ((i+1)%M?' ':'\n');
delete a; cout << endl;

char **c, *pc;
    c = new char*[N]; for(int n=0; n<N; n++) c[n]=new char[M];
    pc = c[0];
    for(int n=0; n<N; n++)
              for(int m=0; m<M; m++) c[n][m] = 65+M*n+m;
    for(int i=0; i<N*M; i++) cout << *(pc+i) << ((i+1)%M?' ':'\n');
    for(int n=0; n<N; n++) { cout << (void*)c[n] << (n==N-1?'\n':'\t'); delete c[n]; }
delete c; cout << endl;

int (*b)[N][M] = new int[K][N][M];
    for(int k=0; k<K; k++)
            for(int n=0; n<N; n++)
                    for(int m=0; m<M; m++)
                    cout << (b[k][n][m] = M*N*k + M*n + m) << (m==M-1?(n==N-1?"\n\n":"\n"):"\t");

    return 0;
}
```

D:\A1\Lect2013\Exmp\4\tst10.exe

```
A B C D
E F G H
I J K L

A B C D
^ ^ ^ ^
^ ^ ^ ^
0x4f0f50        0x4f1240        0x4f1260

0       1       2       3
4       5       6       7
8       9       10      11

12      13      14      15
16      17      18      19
20      21      22      23
```

Компилятор | Ресурсы | Журнал компиляции | Отладка | Результаты поиска | Закрыть

Отладка | Отслеживать | Вывод

Следующий шаг | Продолжить выполнение | Отладка | Добавить в наблюдаемы

Шаг внутрь | Выполнить до курсора | Остановить выполнение | Удалить объект наблюдени

28: 8 | Вставка | 30 строк в файле

D:\Disk_S\Lect2016\Exmp\4\tst10.c - Dev-C++ 5.11

Файл Правка Поиск Вид Проект Выполнить Сервис AStyle Окно Справка

TDM-GCC 4.9.2 64-bit Release

(globals)

Проект Классы O tst10.cpp tst10.c

```c
#include <stdio.h>
#include <malloc.h>

int main() {
const int K=2, N=3, M=4;
int (*b)[N][M] = (int(*)[N][M])calloc(K*N*M,sizeof(int)), k, n, m;
    for(k=0; k<K; k++)
            for(n=0; n<N; n++)
                    for(m=0; m<M; m++)
                    printf("%d%s",(b[k][n][m] = M*N*k + M*n + m), (m==M-1?(n==N-1?"\n\n":"\n"):"\t") );

    return 0;
}
```

D:\Disk_S\Lect2016\Exmp\4\tst10.exe

```
0       1       2       3
4       5       6       7
8       9       10      11

12      13      14      15
16      17      18      19
20      21      22      23


--------------------------------
Process exited after 2.232 seconds with return value 0
Для продолжения нажмите любую клавишу . . .
```

Компилятор Ресурсы Журнал компиляции Отладка Результаты поиска

Line: 6 Col: 59 Sel: 0 Lines: 13 Length: 370 Вставка Done parsing in 0,015 seconds

2_1-2 Составные Т... tst10.c - Dev-C++ D:\Disk_S\Lect2016... D:\Disk_S\Lect2016... ENG 11:45 28.09.2016

Операции Инкремента и Декремента

1. `*++Ptr // *(Ptr += 1)` – **перевод указания на следующий элемент с использованием его значения при вычислении выражения**
2. `--*Ptr // *Ptr -= 1` – **уменьшение текущего элемента на единицу с использованием этого значения в вычислении выражения**
3. `*Ptr++ // (t=Ptr,Ptr+=1,*t)` – **использование значения текущего элемента и перевод указания на следующий элемент**
4. `(*Ptr)-- // (t=*Ptr,*Ptr-=1,t)` – **использование в вычислении текущего значения элемента с последующим его уменьшением на единицу**

D:\A1\S\errors\tst3.c - [Executing] - Dev-C++ 5.10

Файл Правка Поиск Вид Проект Выполнить Сервис AStyle Окно Справка

TDM-GCC 4.8.1 32-bit Debug

(globals)

Проект Класс

f(int a) : int
main() : int

tst3.c

```c
#include <stdio.h>

int f(int a) {
    printf("f::a=%d\n",a);
    return a;
}

int main() {
int *a = (int[]){10,20,30}; int b;
    b = f(*++a) + f(--*a) + f(*a--) + f((*a)++);
    printf("b=%d a[]= %d %d %d\n",b,a[0],a[1],a[2]);

    return 0;
}
```

D:\A1\S\errors\tst3.exe

```
f::a=20
f::a=19
f::a=19
f::a=10
b=68 a[]= 11 19 30

--------------------------------
Process exited after 0.07714 seconds with return value 0
Для продолжения нажмите любую клавишу . . .
```

Компилятор | Ресурсы | Журнал компиляции | Отладка | Результаты поиска

Прервать

Shorten compiler paths

```
Compilation results...
--------
- Errors: 0
- Warnings: 0
- Output Filename: D:\A1\S\errors\tst3.exe
- Output Size: 121,2080078125 KiB
- Compilation Time: 0,91s
```

Line: 1 Col: 1 Sel: 0 Lines: 14 Length: 246 Вставка Done parsing in 0,015 seconds

EN 23:40

D:\A1\S\errors\tst3.cpp - [Executing] - Dev-C++ 5.10

Файл Правка Поиск Вид Проект Выполнить Сервис AStyle Окно Справка

TDM-GCC 4.8.1 32-bit Debug

(globals)

Проект Класс

f(int a) : int
main() : int

tst3.cpp

```cpp
#include <iostream>
using namespace std;

int f(int a) {
    cout << "f::a=" << a << endl;
    return a;
}

int main() {
int a[]={10,20,30},b, *Pa=&a[0];
    b = f(*++Pa) + f(--*Pa) + f(*Pa++) + f((*Pa)--);
    cout << "b=" << b << " a[]=" << a[0] << ' ' << a[1] << ' ' << a[2] << endl;

    return 0;
}
```

Компилятор | Ресурсы | Журнал компиляции | Отладка | Результаты поиска

Прервать

Shorten compiler paths

```
Compilation results...
--------
- Errors: 0
- Warnings: 0
- Output Filename: D:\A1\S\errors\tst3.exe
- Output Size: 1,31301784515381 MiB
- Compilation Time: 1,86s
```

D:\A1\S\errors\tst3.exe

```
f::a=20
f::a=19
f::a=19
f::a=30
b=88 a[]=10 19 29

--------------------------------
Process exited after 3.246 seconds with return value 0
Для продолжения нажмите любую клавишу . . .
```

Line: 12 Col: 72 Sel: 0 Lines: 15 Length: 305 Вставка Done parsing in 0,031 seconds

EN 23:22

Пример 2: операции со строками

- **Вычисление длины строки** (`strlen(a)`)*)

```
char *b=a; while(*b++); return b-a-1;
```

- **Копирование** (`strcpy(a,b)`)

```
while(*a++ = *b++);
```

- **Сравнение** (`strcmp(a,b)`)

```
while(*a++ == *b++) if(!*(a-1)) return 0;
return *(a-1) - *(b-1);
```

- **Конкатенация** (`strcat(a,b)`)

```
char a[N]={…,'\0'},*p=a;
while(*p)p++; while(*p++=*b++);
```

*) `#include <string.h>` - **библиотечные функции Си**

Основные функции ввода/вывода

- `int getchar(void);`
- `int getche(void);`
- `int getch(void);`
- `int [f]getc(FILE*);`

- `int putchar(int);`
- `int putch(int);`
- `int [f]putc(int, FILE*);`

- `int printf("%s", char*);`
- `int puts(const char*);`
- `int fputs(char*, FILE*);`

- `int scanf("%s", char*);`
- `char* gets(char*);`
- `char* fgets(char*, int, FILE*);`

stdin
stdout
stderr

Вывод на экран и в поток

```
#include <stdio.h>
#include <conio.h>
#include <string.h>

int main() {const char *s1="This
   is output into the stdout
   stream",
   *s="to the console";
char *p, s2[]="This is output into
   the stdout stream";
//    *s1='a';
while(*s1) putchar(*s1++);
   putchar('\n');
 p=strstr(s2,"stdout");
*(p+3)='e'; *(p+5)=*(p+4)='r';
p=s2;
while(*p) fputc(*p++,stderr);
fputc('\n',stderr);

p=strstr(s2,"into");
while(*p++=*s++); p=s2;
while(*p) putch(*p++);
putch('\n');

return 0;
}
```

Вывод

C:\Windows\System32\cmd.exe

```
D:\A1\Lect2013\Exmp\4>a >out 2>err
This is output to the console
```

```
#include <s
#include <conio.h>
#include <string.h>

int main() {const cha
   is output into the
   stream",
   *s="to the console";
char *p, s2[]="This is output i
   the stdout stream";
//    *s1='a';
while(*s1) putchar(*s1++);
   putchar('\n');
```

err — Блокнот

Файл Правка Формат Вид Справка

This is output into the stderr stream

out — Блокнот

Файл Правка Формат Вид Справка

This is output into the stdout stream

Стр 1, стлб

```
*(p+3)='e'; *(p+5)=*(p+4)='r';
p=s2;
utc(*p++,stderr);
derr);
s2;
);
putch('\n');

return 0;
}
```

Dev-C++ 4.9.9.2

Файл Правка Поиск Вид Проект Выполнить Отладка Сервис CVS Окно Справка

tst02_1.c

```c
#include <stdio.h>

char* getstr(char* s, FILE* a) {
int b=1, l;
char c[10];
     if(feof(a)) return NULL;
     s=(char*)calloc(1,1);
     while(b && fgets(c,10,a)) {
           if(c[(l=strlen(c))-1]=='\n') { b=0; c[l-1]='\0'; }
           s = (char*)realloc(s,strlen(s)+(b?l:l-1)+1);
           strcat(s,c); }
     return s;
}

int main() {
char* s;
      while(s=getstr(s,stdin)) {
            puts(s); printf(" %d\n",strlen(s));
            free(s); }
      return 0;
}
```

Компилятор | Ресурсы | Журнал компиляции | Отладка | Результаты поиска

1: 1 | Вставка | Ready.

Dev-C++ 4.9.9.2

Файл Правка Поиск Вид Проект Выполнить Отладка Сервис CVS Окно Справка

tst02_1.c

```c
#include <stdio.h>

char* getstr(char* s, FILE* a) {
int b=1, l;
char c[10];
      if(feof(a)
      s=(char*)c
      while(b &&
            if(c
            s =
            strc
      return s;
}

int main() {
char* s;
      while(s=getstr(s,stdin)) {
            puts(s); printf(" %d\n",strlen(s));
            free(s); }
      return 0;
}
```

D:\A1\Lect2013\Exmp\4\tst02_1.exe

```
sasasa sasasasa asasasasa asasasasas asasasasa asasasa asasas assasasass   t
sasasa sasasasa asasasasa asasasasas asasasasa asasasa asasas assasasass   t
 76
asaas sdfdfdf sdsdsds dsdsdsd fdfdfddf
asaas sdfdfdf sdsdsds dsdsdsd fdfdfddf
 38


 0
vvvv
```

Dev-C++ 4.9.9.2

Файл Правка Поиск Вид Проект Выполнить Отладка Сервис CVS Окно Справка

[*] tst03.cpp

```
#include <iostream>
using namespace std;

istream& operator>> (istream& b, char* &a) {
int l=0,la=0;
const int n=10;
char s[n]; s[n-1]='\0'; *(a = new char)='\0';
    if(b.peek()==EOF) return b;
    while(b.get(s[l]),s[l++]!='\n')
        if(l==n-1) { l=0; a=(char*)realloc(a,la+=n); strcat(a,s); }
    if(l>1) { s[l-1]='\0'; a=(char*)realloc(a,la+l); strcat(a,s); }
    return b;
}

int main() {
char *s;
     while(cin >> s) {
        cout << s << ' ' << strlen(s) << endl;
        delete s; }
    return 0;
}
```

Компилятор | Ресурсы | Журнал компиляции | Отладка | Результаты поиска

Dev-C++ 4.9.9.2

Файл Правка Поиск Вид Проект Выполнить Отладка Сервис CVS Окно Справка

[*] tst03.cpp

```
#include <iostream>
using namespace std;

istream& operator>> (istream& b, char* &a) {
int l=0,la=0;
const int n=10;
char s[n]; s[n-1]='\0'; *(a = new char)='\0';
    if(b.peek()==EOF) return b;
    while(b.get(s[l]),s[l++]!='\n')
        if(l==n-1) { l=0; a=(char*)realloc(a,la+=n); strcat(a,s); }
    if(l>1) { s[l-1]='\0'; a=(char*)realloc(a,la+l); strcat(a,s); }
    return b;
}

int main() {
char *s;
     while(cin >>
        cout << s
        delete s;
    return 0;
}
```

D:\A1\Lect2013\Exmp\4\tst03.exe

```
qwqwqwqwqwq qwqwqwqw qwqwqwqwq qwqwqwqwq qwqwqwqwqw
qwqwqwqwqwq qwqwqwqw qwqwqwqwq qwqwqwqwq qwqwqwqwqw 51
asasas          dddddd
asasas          dddddd 21
aaaa
```

Пример 3: потеря быстродействия

- **Для заданных строк s1 и s2 найти номер символа, с которого начинается вхождение строки s2 в строку s1.**
- **int strncmp(char* s1, char* s2, int n) – производит сравнение не более n символов из двух строк и возвращает значение 0 – только в случае если строки совпадают.**
- **int main(int argn, char* argv[ ])**
  - argn – количество слов в командной строке
  - argv – параметры, переданные программе

Dev-C++ 4.9.9.2

Файл Правка Поиск Вид Проект Выполнить Отладка Сервис CVS Окно Справка

Создать Вставить Переключить Перейти

Проект Классы Отладка

tst4.c

```c
#include <stdio.h>
#include <string.h>

unsigned long long start, end;
unsigned long long access_counter() { asm("rdtsc"); }

int main(int argc, char *argv[]) {
int i,l1,l2;
    if(argc<3) { printf("Error parameter list\n"); return 1; }
    start=access_counter();
    for(i=0; i<strlen(argv[1]); i++)
            if(!strncmp(argv[1]+i,argv[2],strlen(argv[2])))
                    printf("%s %s %d\n",argv[1],argv[2],i);
    end=access_counter();
    printf("%u\n",(unsigned)(end-start));
    start=access_counter();
    l1=strlen(argv[1]); l2=strlen(argv[2]);
    for(i=0; i<l1; i++)
            if(!strncmp(argv[1]+i,argv[2],l2))
                    printf("%s %s %d\n",argv[1],argv[2],i);
    end=access_counter();
    printf("%u\n",(unsigned)(end-start));
    return 0;
}
```

Компилятор Ресурсы Журнал компиляции Отладка Результаты поиска Закрыть

Отладка Отслеживать Вывод

Следующий шаг Шаг внутрь Продолжить выполнение Выполнить до курсора Отладка Остановить выполнение Добавить в наблюдаемы Удалить объект наблюдени

23: 6 Вставка 25 строк в файле

D:\A1\Lect2013\Exmp\4\tst4.exe

```
1234567890123456789 0*12*1234567890 *12* 20
335447
1234567890123456789 0*12*1234567890 *12* 20
121051
```