

# Въведение в програмирането

## Езици за програмиране

Някои изброяват 8000 езика, други – 2500 (а останалото – "диалекти")

## Поколения

1-во – машинен код

2-ро – асемблер

3 - т о – с о б щ о п р е д н а з н а ч е н и е  
(преобладаващите)

4-то – със специално предназначение – БД,  
отчети, интерфейс и др.

5-то – проблемно-ориентирани, повечето за  
изкуствен интелект

Локални приложения		Мобилни приложения	Уеб приложения
големи, сложни	по-прости, прототипи		
C/C++  (Objective C за Mac OS)	C#, Visual BASIC	Java, JavaScript, C++, C#, Swift	Java, JavaScript, C#, Python, PHP, Perl

<http://www.lextrait.com/Vincent/implementations.html>

Език	Предмет
C/C++	Въведение в програмирането, задължителен, 1 курс
C++	Обектно-ориентирано програмиране, задължителен, 2 курс + Visual C++, избираем, 3 курс
C#	Програмиране и структури от данни, задължителен, 2 курс + .NET технологии, задължителен, 3 курс
Java	Java, задължителен, 3 курс
JavaScript	Уеб технологии, задължителен, 3 курс
PHP, още JavaScript	Сървърно програмиране, задължителен, 4 курс

<http://informatics.ue-varna.bg/olimpiada/index.php>

C:

1969-1973 Денис Ричи, Брайън Кернигън

C++:

Bjarne Stroustrup 1983-1985

Предимства на C/C++:

- скорост на изпълнение на кода
- пълна поддръжка на възможностите на операционната система
- език от високо ниво

**Среди за програмиране, компилатори:**

Visual Studio / Visual C++

Borland C++ (Builder)

GCC

Dev-C++

Djgpp

Intel C++

# Въведение в езика C/C++, структура на програмата

прави се разлика малки / главни букви

if If IF iF

a A

```
#include "pch.h"
```

```
#include <iostream>
```

```
int main()
```

```
{
```

```
    std::cout << "Hello World!\n";
```

```
}
```

```
#include "pch.h"
```

```
#include <iostream>
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    cout << "Hello World!\n";
```

```
}
```

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

```
int main()
{
    int a;
    cin>>a;
    a=a+1;
    cout<<a;
}
```

# Типове данни и операции (част 1)

Цели числа:

char	1	-128	+127
short int	2	-32768	+32767
int	4	-2 млрд.	+ 2 млрд.
long int	4	-2 млрд.	+ 2 млрд.
long long	8	няма го навсякъде –9223372036854775808 до 9223372036854775807	

```
short int f;  
short f;  
long Dalgo_chislo987;
```

```
unsigned char c;  
unsigned short int i;  
unsigned long int ll;  
unsigned short k;  
unsigned u;    // същото като unsigned int u;
```

```
int j;  
j=5;  
j++;    j=j+1;    j--;
```

```
int j=5;
```

```
int a;
```

```
a++;
```

```
int a, b=4, c=13;
```

```
/*
```

```
    a=5;
```

```
    произволен текст
```

```
    dssdfsd коментар
```

```
*/
```

```
int a; // това е пром. за цената a=5;
```

Извеждане на кирилица в конзолата:

```
setlocale(LC_ALL, "");
```

# Управляващи оператори (част 1)

if(условие) действие;

```
int oценка, vzeti_izpiti=2;  
cin>>oценка;  
if(oценка>2) vzeti_izpiti++;  
cout<<vzeti_izpiti;
```

проверка равно == или различно !=

```
int oценка;  
cin>>oценка;  
if(oценка==2) cout<<"Слаб";
```

```
int oценка;  
cin>>oценка;  
if(oценка!=2) cout<<"Не е слаб";
```

```
int a, b=4;  
cin>>a;  
if(a) b++;    // същото като if(a!=0) b++;  
cout<<b;
```



```
int a, b=4;  
cin>>a;  
if(!a) b++; // същото като if(a==0) b++;  
cout<<b;
```

логическо "и" && и "или" ||

```
int oценка;  
cin>>oценка;  
if(oценка>=2 && oценка<=6) cout<<"OK";
```

```
int oценка;  
cin>>oценка;  
if(oценка<2 || oценка>6) cout<<"Not OK";
```

```
int a;  
cin>>a;  
if(a<3 && a>8) ... // никога няма да е вярно  
// условието
```

## ИЗПЪЛНЕНИЕ НА БЛОК

```
int a, b, c;  
cin>>a;  
if(a==5)  
{  
    b=3;  
    c=7;  
}
```

```
int a, b, c;  
cin>>a;  
if(a==5) b=3; c=7;
```

if(условие) действие1; else действие2;

```
int oценка;  
cin>>oценка;  
if(oценка>=2 && oценка<=6) cout<<"OK";  
else cout<<"Not OK";
```

```
if(a>3) b=4;  
else {b=5; c=3;}
```

```
if(a>3) b=4;  
else if(a>1) c=7;  
else c=3;
```

```
if(a>1) b=4;  
else if(a>3) c=7;  
else c=3;
```

ЦИКЪЛ

```
while(условие) действие;
```

```
int a=1;  
while(a<=10)  
{  
    cout<<a;  
    a++;  
}
```

```
int a=4;  
while(a)  
{  
    cout<<a;  
    a--;
```

```
}
```

```
do
```

```
    действие;
```

```
while(условие);
```

```
int a=4;
```

```
while(a)
```

```
{
```

```
    cout<<a;
```

```
    a--;
```

```
}
```

```
int a=4;
```

```
do {
```

```
    cout<<a;
```

```
    a--;
```

```
} while(a);
```

```
int a;
```

```
cin>>a;
while(a)
{
    cout<<a;
    a--;
}
```

```
cin>>a;
do {
    cout<<a;
    a--;
} while(a);
```

```
int oценка;
do {
    cout<<"Въведете оценка (2-6):";
    cin>>оценка;
} while(оценка<2 || оценка>6);
cout<<"OK\n";
```

```
int a=1, s=0;
while(a<=10)
{
    s+=a;
    a++;
}
cout<<s;
```

```
for(начало;условие;стъпка)  
    действие;
```

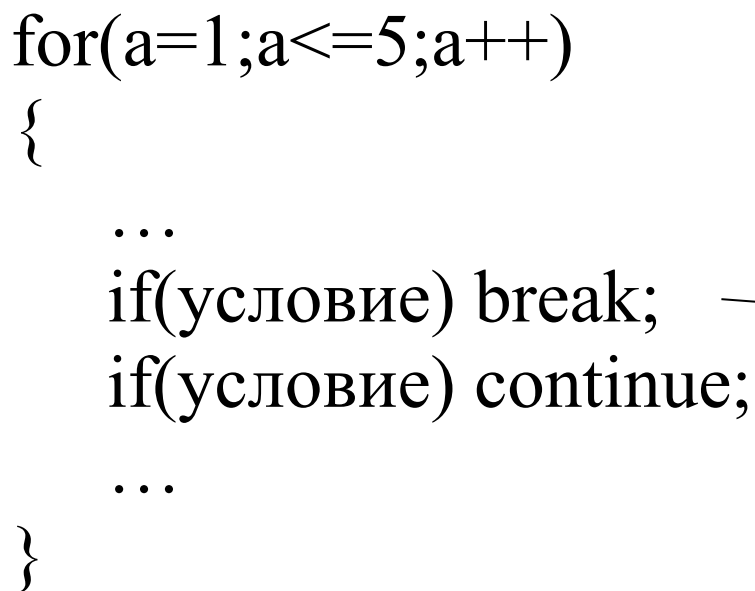
```
int i;  
for(i=0; i<10; i++)  
{  
    cout<<i;  
}
```

```
for(a=2;a<10;a+=3)  
    cout<<a;
```

```
for(a=2; b<8; c++)  
{  
    ...  
}
```

```
for(a=2, b=3; a<10; a+=3, b++)  
    cout<<a<<" "<<b<<"\n";
```

```
for(a=1;a<=5;a++)  
{  
    ...  
    if(условие) break;  
    if(условие) continue;  
    ...  
}
```



```
int a, b;  
for(a=1;a<=5;a++)  
{  
    cout<<"Изпит: "<<a<<"\n";  
    cout<<"Въведете оценка: ";  
    cin>>b;  
    if(b==2)  
    {  
        cout<<"Играта свърши\n";  
        break;  
    }  
    cout<<"Все още сте студент\n";  
}
```

```
int a, b;
for(a=1;a<=5;a++)
{
    cout<<"Изпит по математика: "<<a<<"\n";
    cout<<"Въведете точки от практиката: ";
    cin>>b;
    if(b<40)
    {
        cout<<"Не стигате до теория\n";
        continue;
    }
    cout<<"Явявате се на теория\n";
}
```



```
switch (a)
{
    case 3:
        b=7;
        break;
    case 1:
        c=8;
        break;
    default:
        c=11;
}
```

```
if(a==3) b=7;
else if(a==1) c=8;
else c=11;
```

```
switch(a)
{
    case 7:
    case 4:
    case 11:
        b=3;
```

```
        c=7;
        break;
case 5:
    b=1;
    c=3;
    break;
default:
    d=8;
}
```

```
int oценка;
cin>>oценка;
switch (oценка)
{
    case 2:
        cout<<"Слабак";
        break;
    case 5:
    case 6:
        cout<<"Получава стипендия";
    case 3:
    case 4:
        cout<<"Успешно положен изпит";
        break;
```

default:

```
    cout<<"Невалидна оценка";  
}
```

условие?стойност1:стойност2

max=a>b?a:b;

if(a>b) max=a; else max=b;

оператор goto

int oценка;

пак:

cout<<"Въведете оценка (2-6):";

cin>>oценка;

if(oценка<2 || oценка>6) goto пак;

cout<<"OK\n";

int oценка;

do {

cout<<"Въведете оценка (2-6):";

cin>>oценка;

} while(oценка<2 || oценка>6);

cout<<"OK\n";

## Типове данни и операции (част 2)

float 4

double 8

```
double uspeh;
```

```
uspeh=3.45;
```

```
const int a=4;
```

```
const double dds=0.2;
```

```
double f;
```

```
cin>>f;
```

```
cout<<f*(1+dds); // вариант 1
```

```
cout<<f+f*dds; // вариант 2
```

```
cout<<5/3;
```

```
cout<<5/3.0;
```

```
cout<<(float)5/3;
```

```
cout<<a/b.0; // ГРЕШКА!!!
```

```
cout<<(float)a/b;
```

```
cout<<a*1.0/b;
```

```
a=(b*(c+2)-3)*d;
```

a++;

a--;

++a;

int a=5;

a++;

a--;

++a;

int a=2, b;

b=a++;

b=++a;

a=3 b=2

a=3 b=3 // това се получава

a\*=4; a=a\*4;

c/=b; c=c/b;

d+=5; d=d+5;

promenliva-=10;

promenliva=promenliva-10;

int a=19, b=4, c, d;

c=a/b;

d=a%b;

$40/6=?$

$40\%6=?$

```
int c, d, a=5, b=3;  
c=sizeof(a)-sizeof(b);  
d=a-b;
```

# Масиви и стрингове

```
int oценка[10];
```

```
oценка[0]=4;
```

```
...
```

```
oценка[9]=6;
```

```
oценка[10]=2; // НЕ!!!
```

```
int oценка[10];
```

```
for(int i=0; i<10; i++)
```

```
    cin>>oценка[i];
```

```
int gladnici[5], sum=0;
```

```
for(int i=0; i<5; i++)
```

```
{
```

```
    cout<<"Колко понички е изял гладник  
№"<<i+1<<": ";
```

```
    cin>>gladnici[i];
```

```
    sum+=gladnici[i];
```

```
}
```

```
cout<<"Общо гладниците са изяли  
"<<sum<<" понички\n";
```

```
int b[100][40];
```

```
b[0][0]=6;
```

```
...
```

```
b[4][7]=3;
```

```
...
```

```
b[99][39]=2;
```

```
for(int i=0; i<100; i++)
```

```
    for(int j=0; j<40; j++)
```

```
        cin>>b[i][j];
```

```
for(int i=0; i<100; i++)
```

```
{
```

```
    cout<<"Започваме да въвеждаме оценките  
на студент "<<i+1;
```

```
    for(int j=0; j<40; j++)
```

```
{
```

```
    cout<<"Въведете оценка "<<j+1;
```

```
    cin>>b[i][j];
```

```
}
```

```
}
```

ASCII код



```
char g;  
g='A';  
g++;  
g='C'-'A';
```

```
char s[10];  
s[0]=4;  
s[1]=6;  
s[2]='Z'; // s[2]=90;
```

V	a	r	n	a		j	l	b	f
86	97	114	110	97	0	106	108	98	102

```
char s[10];  
s="Varna"; //грешно  
strcpy_s(s, "Varna");
```

```
char s[10]="Varna";  
s[1]='e';  
s[3]=0;  
s[1]='0';
```

```
#include <string.h>
#include <stdlib.h>
```

Много от стандартните функции не са безопасни и в C++ са заменени с по-сигурни версии с `_s` накрая, напр. `strcpy` -> `strcpy_s`, `gets` -> `gets_s` и т.н.

```
if(s=="Kuku") // грешно
strcmp(s1,s2)
```

```
if(strcmp(s, "Kuku")==0) cout<<"Стрингът е
равен на kuku";
```

```
if(!strcmp(s, "Kuku")) cout<<"Стрингът е
равен на kuku";
```

```
strcmp("abc", "cab") // -1
strcmp("fff", "bbb") // 1
strcmp("abd", "aae") // 1
```

```
char s[40];
cin>>s;
```

```
gets_s(s);
```

```
strlen(s);
```

```
char s[20];
```

```
strcpy_s(s, "Test");
```

```
cout<<strlen(s);    // 4
```

```
strcpy_s(s, "Great test #@!");
```

```
cout<<strlen(s);    // 14
```

```
// strcpy_s(s1, s2);
```

```
char s1[5], s2[10];
```

```
strcpy_s(s2, "Crocodile");
```

```
strcpy_s(s1, s2); // грешка
```

```
strncpy_s(s1, s2, n);
```

```
char s[4];
```

```
strncpy_s(s, "Varna", 7); // грешка
```

```
strncpy_s(s, "Varna", 3);
```

```
char s1[10];
```

```
strncpy_s(s1, "Varna", 9); // OK
```

```
strncpy_s(s, s1, 3);
```

```
strcmp(s1, s2)
```

```
strncmp(s1, s2, n)
strcmp("Varna", "Varka")
strncmp("Varna", "Varka", 3)
strncmp("Do", "Donkey", 3)
```

```
strcmp(s1, s2);
strcmp("Varna", "varna"); //не равни
strcmp("Varna", "varna"); //равни
```

```
strnicmp(s1, s2, n);
strnicmp("Varna", "varka", 3); //равни
```

```
// strcat_s(s1, s2);
char s[30];
strcpy_s(s, "2770");
strcat_s(s, " is the best");
// 2770 is the best
```

```
//strncat_s(s1, s2, n);
strcpy_s(s, "2603");
strncat_s(s, " is very good", 6);
cout<<s; // 2603 is ve
```

## Вход/изход с printf/scanf и форматиране на изхода с printf

printf

scanf

// scanf не се препоръчва, понеже не е безопасна

// да се ползва scanf\_s – обезопасен вариант

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

```
int a;
```

```
scanf_s("%i", &a);    // cin>>a;
```

```
a++;
```

```
printf("%i", a);      // cout<<a;
```

```
int a,b;
```

```
scanf_s("%i%i", &a, &b);    // cin>>a>>b;
```

```
a=4;
```

```
printf("Получената оценка на изпита е = %i",  
a);
```

```
Получената оценка на изпита е = 4
```

```
int oценка;  
scanf_s("%i", &оценка);  
printf("Каква е тази оценка %i, която си  
получил?!\\n", оценка);
```

```
int a=5, b=3;  
printf("%i не е %i", b, a); // 3 не е 5
```

Форматиращи символи:

%c – символ (char), изисква брой (1)

%d, %i – десетично цяло число int

%u – десетично цяло неотрицателно число int

%f – реално число

%e, %E – "научен" формат

%g, %G – използва %e или %f, което е по-кратко

%x, %X – int в шестнадесетичен формат

%s – стринг

%% – извежда %

Модификатори (към d,i,u,x)

l – long

h – short, напр.:

%li = long int

%hu = unsigned short int

\n = нов ред

\t = табулация

```
char k;  
scanf_s("%c", &k, 1);  
printf("%i", k);
```

```
char k;  
for(k='A';k<='Z';k++)  
    printf("%c има код %i\t", k, k);
```

```
int a;  
char s[20];  
scanf_s("%i", &a);  
scanf_s("%s", s, 20); // брои се и 0-та
```

```
float f=3.2, m=20;  
printf("%f\n%f", f, m);  
3.20000  
20.00000
```

```
printf("%8.2f\n%8.2f", f, m);  
    3.20  
    20.00
```

```
printf("%8d\n", 100); // right-justified
printf("%-8d\n", 100); // left-justified
```

100

100

```
printf("%d%d", 100, 100);
```

100100

```
printf("%-8d%d", 100, 100);
```

100 100

\*\*\* до тук е материалът за Тест 1 \*\*\*



# Указатели и динамично заделяне на памет

адрес	Пром.	израз	Ст-ст
10	i	i	7
11	i	&i	10
12	i	*p	7
13	i	p	10
14	p	&p	14
15	p		
16	p		
17	p		

```
int i; i=5;
```

```
int *p;
```

```
p=&i;
```

```
*p=7;
```

```
cout << "i=" << i << endl << "&i=" << &i << endl;
```

```
cout << "*p=" << *p << endl << "p=" << p << endl;
```

```
cout << "&p=" << &p << endl;
```

```
int a=5, b=7; // допускаме, че a и b са
```

```
// последователни в паметта
```

```
int *p=&a;  
*(p+1)=6;
```

```
short *ps;  
ps+6 сочи колко байта по-надолу?
```

```
int a[10];  
&a[0] е еквивалентно на а (името на масива)
```

```
int n;  
cin>>n;  
int a[n]; // грешка!!!
```

```
int n;  
cin>>n;  
int *p;  
p = new int[n];  
p[0]=5;  
p[1]=3;      // *(p+1)=3;  
...  
delete []p;
```

```
int *k;  
k=new int;  
*k=7;  
  
...  
delete k;
```

# Структури

```
struct тип {  
    данни;  
} променливи;
```

```
struct student {  
    char ime[40];  
    int fn;  
    int kurs;  
} ivan, maria;
```

```
ivan.fn=2645;  
strcpy_s(ivan.ime, "Ivan Ivanov");  
cout<<ivan.ime<<" "<<ivan.fn;
```

```
maria.kurs=3;
```

```
student vasil;  
cin>>vasil.fn>>vasil.kurs;  
vasil.fn=222;
```

// вложени структури

```
struct adres {  
    char ulica[30];  
    char grad[20];  
    int pk;  
};
```

```
struct student {  
    char ime[40];  
    int fn;  
    adres adr;  
} ivan;
```

```
ivan.fn = 1000;  
strcpy_s(ivan.ime, "Ivan");  
ivan.adr.pk = 9002;  
strcpy_s(ivan.adr.grad, "Varna");
```

// масив от структури

```
student gr30[15];  
strcpy_s(gr30[0].ime, "John");  
gr30[1].adr.pk=1000;
```

```
int i;  
for(i=0; i<15; i++)  
{  
    printf("Въведете името на човек №%i: ", i+1);  
    gets_s(gr30[i].ime);  
}
```

// указател към структури

```
student *ps;  
ps=&ivan;
```

```
(*ps).fn=1200;  
ps->fn=1200;
```

// динамичен масив от структури

```
student *gr31;
```

```
cin>>n; // тук разбираме колко студента ще има  
gr31 = new student[n];  
gr31[0].fn = 1002;  
...
```

```
gr31[3].fn = 1020;
```

```
...
```

```
delete []gr31;
```

```
student *gr31;
```

```
int i, n;
```

```
scanf_s("%i", &n);
```

```
gr31 = new student[n];
```

```
for (i = 0; i < n; i++)
```

```
{
```

```
    printf("Въведете Ф№ на човек №%i: ", i + 1);
```

```
    scanf_s("%i", &gr31[i].fn);
```

```
    printf("Въведете името на човек №%i: ", i + 1);
```

```
    getchar(); // заради проблемната комбинация
```

```
                // scanf_s и gets_s
```

```
    gets_s(gr31[i].ime);
```

```
}
```

```
// търсене по ФН
```

```
int fn;
```

```
printf("Въведете Ф№ за търсене: ");
```

```
scanf_s("%i", &fn);
for (i = 0; i<n; i++)
    if (gr31[i].fn == fn)
    {
        printf("%s\n", gr31[i].ime);
        break;
    }
```

// многомерен (двумерен) масив от  
структури

```
student inf[4][25];
inf[0][11].fn=1021;
```

// търсене по Ф№, ако са запълнени

```
int i, j, fn;
cin>>fn;
for(i=0; i<4; i++)
    for(j=0; j<25; j++)
        if(gr31[i][j].fn==fn)
        {
            cout<<gr31[i][j].ime;
            break;
        }
```



}

# Потребителски функции

```
тип име(тип1 параметър1, ...)  
{  
    тяло (код)  
}
```

```
int saberi(int a, int b)  
{  
    int c;  
    c=a+b;  
    return c;  
}
```

```
int main()  
{  
    int b;  
    b=saberi(3, 7)*6;  
    cout<<b;  
    return 0; }
```

```
int saberi(int a, int b);  
int main()  
{  
    int b;  
    b=saberi(3, 7)*6;  
    cout<<b;  
    return 0;  
}
```

```
int saberi(int a, int b)  
{  
    int c;  
    c=a+b;  
    return c; }
```

```
void f1(int a)
{
    if(a<3) return;
    cout<<"Получената оценка е "<<a;
}
```

```
f1(6);
f1(2);
```

// не!!! без вложени функции

```
void f1(int a)
{
    ....
    void f2(int b)
    {
        ...
    }
    ...
}
```

```
int data(void)
{ // допускаме, че функцията намира
  // коя дата от месеца сме в n
  ...
  return n;
}
```

```
int data()
{ // допускаме, че функцията намира
  // коя дата от месеца сме в n
  ...
  return n;
}
```

```
void pozdrav(void)
{
    cout<<"Здравейте, дами и господа!\n";
}
```

```
void pozdrav()
{
    cout<<"Здравейте, дами и господа!\n";
}
```

```
f1()
{
    ...
    return n;
}
```

```
int validna(int a)
{
    if(a>=2 && a<=6) return 1;
    else return 0;
}
```

това е в main:

```
int oценка;  
cin>>oценка;  
if(validna(oценка)) cout<<"OK";  
else cout<<"Невалидна оценка";
```

```
int oценка;  
do {  
    cin>>oценка;  
} while (!validna(oценка));
```

```
int c;  
int f1()  
{  
    int b;  
    ...  
}  
int main()  
{  
    int a;  
    int b;  
    ...  
}  
int f2()  
{  
    ...  
}
```



```

int k;
void f1() {
    int k;
    k=7;
    ::k=6;
}

void f2() {
    int i;
    cout<<k;
}

int main() {
    int j;
    k=3;
    f1();
    f2();
    cout<<k;
    return 0;
}

```

// преговор указатели

```

int i=56;
int *p=&i;
*p=8;

```

// псевдонимы (или референции)

```

int i=56;
int &a=i;
a=8;

```

```
void f1(int a)
{
    a=5000;
    cout<<a;
}
```

```
void f2(int *a)
{
    *a=5000;
    cout<<*a;
}
```

```
void f3(int &a)
{
    a=5000;
    cout<<a;
}
```

```
int main()
{
    int zap=3000;
    f1(zap);
    cout<<zap;
    return 0;
}
```

```
int main()
{
    int zap=3000;
    f2(&zap);
    cout<<zap;
    return 0;
}
```

```
int main()
{
    int zap=3000;
    f3(zap);
    cout<<zap;
    return 0;
}
```

# Математически функции

`abs(x)` – абсолютна стойност

`sin(x)`, `cos(x)`, `tan(x)`, `asin(x)`, `acos(x)`, `atan(x)`

`atof(s)` – от стринг в double

```
char s[10]="3.14";
```

```
// double a=s+5; // НЕ!!!!!!!
```

```
double a=atof(s)+5;
```

`atoi(s)`

`atol(s)`

```
char s[10]="3";
```

```
int i=atoi(s);
```

`itoa(value, string, radix)`

ltoa(value, string, radix)

```
int i=5;
```

```
char s[20];
```

```
// оригинална функция itoa, но заменена с  
_itoa и безопасен вариант _itoa_s
```

```
_itoa_s(i, s, 10);
```

```
cout<<s; // 5
```

```
_itoa_s(i, s, 2);
```

```
cout<<s; // 101
```

```
_itoa_s(31, s, 16);
```

```
cout<<s; //1f
```

floor(x) – по-малкото цяло число

ceil(x) – по-голямото цяло число

```
cout<<floor(2.8); // 2
```

```
cout<<floor(-2.2); // -3
```

```
cout<<ceil(2.2); // 3
```

```
cout<<ceil(-2.8); // -2
```

$\exp(x)$  – e на степен x

$\text{pow}(x, y)$  – x на степен y

```
cout<<pow(3, 2); // 9
```

// "ръчно" повдигане на цяла степен

```
int a, b;
```

```
cin >> a >> b;
```

```
int i, result=1;
```

```
for (i = 1; i <= b; i++)
```

```
    result *= a;
```

```
cout << result;
```

```
sqrt(x)
```

```
cout<<sqrt(6); // гледай си работата
```

## Функции, свързани със символи

```
isalpha(x)
```

```
char c='a';
```

```
if(isalpha(c)) ...
```

```
// нещо – ако е буква от A-Z a-z
```

```
char s[20]="This 87.89 is a text!";
```

```
if(isalpha(s[2])) ...
```

```
// колко символа от въведен стринг са букви  
от азбуката
```

```
char s[20];
```

```
gets_s(s);
```

```
int i, count = 0;
```

```
for (i = 0; i < strlen(s); i++)
```

```
    if (isalpha(s[i])) count++;
```

```
cout << count;
```

`islower(x)` // дали е от a-z

`isupper(x)` // дали е от A-Z

`isdigit(x)` // дали е цифра

`isalnum(x)` // дали е буква или цифра

`ispunct(x)` // дали е пунктуационен символ

`isspace(x)` – ASCII 9-13, 32

`c2=tolower(c1);`

`c2=toupper(c1);`

`char c;`

`cin >> c;`

`cout << (char)toupper(c);`



## Заделяне на памет

```
int n;  
cin>>n;  
int a[n]; // не става!!!
```

```
int n;  
cin>>n;  
int *p;  
p = new int[n];  
p[0]=5;  
p[1]=3;  
...  
delete []p;
```

// функция за заделяне на памет malloc(общ размер)

```
cin>>n;  
int *p;  
p=(int*)malloc(20*sizeof(int));  
p[0]=5;  
p[1]=3;  
...
```

// функция за заделяне на памет calloc(бройка,  
размер)

p=(int\*)calloc(20, sizeof(int));

// функция за преоразмеряване realloc(стар указател,  
нов размер в байтове)

p=(int\*)realloc(p, 15\*sizeof(int));

// функция за освобождаване на паметта  
free(указател)

free(p);

# Процеси

exit(код на завършване)

exit(0);

system("програма за изпълнение")

system("C:\\Dir1\\File.exe");

system("C:/Dir1/File.exe");

system("cls");

// организиране на меню и извикване на програма

int izbor;

do {

cout << "1.Пусни Notepad\\n";

cout << "2.Пусни Calculator\\n";

cout << "3.Изчисти екрана\\n";

cout << "4.Изход\\n";

cin >> izbor;

switch (izbor)

{

case 3:

```
        system("cls");
        break;
    case 1:
        system("notepad.exe");
        break;
    case 2:
        system("calc.exe");
        break;
    }
} while (izbor != 4);
```

\*\*\* до тук е материалът за тест 2

# Допълнителен / незадължителен материал (няма да участва в изпита)

## двоична аритметика

$$237 \text{ (десетично)} = 7 + 3*10 + 2*100$$

$$11011 \text{ (двоично)} = 1 + 1*2 + 0*4 + 1*8 + 1*16 \\ = 27 \text{ (десетично)}$$

$$0101101 \text{ (двоично)} = 1*1 + 0*2 + 1*4 + 1*8 + \\ 0*16 + 1*32 = 45 \text{ (десетично)}$$

&

$$1 \& 4 = 0$$

0001

0100

0000

$$2 \& 2 = 2$$

$$2 \& 3 = 2$$

0010

0011

0010

Шегичка?

Има 10 вида хора – такива, които разбират двоичен код и такива, които не разбират

|

2 | 3

10

11

11

2 | 4

0010

0100

0110

~

unsigned char c=3, d;

d=~c;

00000011

11111100

$\wedge$  // още известно като XOR

0011

0110

0101

```
int a, b;
```

```
cin>>a;
```

```
b=a^a;
```

```
cout<<b;
```

```
int a=3, b;
```

```
b=a<<2;
```

0011

1100

```
b=a>>1;
```

0011

0001

// объединения

```
int a;
```

```
int b;
```

```
float c;  
double d;
```

```
union тип {  
    данни;  
} променливи;
```

```
union some {  
    int a;  
    int b;  
    float c;  
    double d;  
} uv;
```

```
aaaa  
bbbb  
cccc  
dddddddddd
```

```
uv.a = 5;  
...  
uv.c = 3.14; // тук вече a губи стойността си  
...
```

**Предефиниране на функции**  
// overloading, overloaded



```
int max(int a, int b)
{   if(a>b) return a; else return b; }
```

```
int max(int a, int b, int c)
{   if(a>b && a>c) return a;
    else if(b>c) return b;
    else return c; }
```

```
void main()
{
    int c, x, y, z;
    cin>>x>>y>>z;
    c=max(x, y);
    c=max(x, y, z);
}
```

```
// собствени хедър файлове при проекти  
// с много файлове код
```

```
#include "somefile.h"  
#include <somefile.h>
```

```
A.CPP  
void f1()  
{ ... }  
void f2()  
{ ... }
```

```
A.h  
extern void f1();  
extern void f2();
```

```
B.CPP  
#include "A.h"  
int main()  
{  
    f1();  
    ...  
    f2();  
}
```

# Модификатори за съхранение на променливите

## storage modifiers

```
//модификатор auto  
auto int i;
```

```
auto int j; // грешка!  
void main()  
{  
    auto int i;  
}
```

```
//модификатор extern
```

```
a.cpp:  
int a=3;  
// код  
void f1()  
{ a++; cout<<a;}
```

```
b.cpp:  
extern void f1();  
extern int a;  
void main()  
{  
    a=6; f1();  
}
```

//модификатор static  
локални:

```
void f1()
{
    int a=0;
    a++;
    cout<<a;
}
```

```
int main()
{
    f1();
    f1();
    return 0;
}
```

```
void f1()
{
    static int a=0;
    a++;
    cout<<a;
}
```

```
int main()
{
    f1();
    f1();
    return 0;
}
```

глобални // долното ще даде грешка

a.cpp:

```
static int a;
// КОД
```

b.cpp:

```
extern int a;
void main()
{
    a=6;
}
```

```
//модификатор register  
register int i;
```

```
// целочислено деление и остатък
```

Стандартна структура `div_t` с полета:  
`quot, rem`

```
div_t result;  
result = div(10, 3);  
cout<<result.quot;    // 3  
cout<<result.rem;     // 1
```

```
int q, r;  
q=10/3;  
r=10%3;
```

## **Работа с дати/часове**

```
#include <time.h>
```

`time_t` – брой секунди от 1.01.1970 г. (long)

```
struct tm {  
    int tm_sec;    /* секунди, 0-59 */  
    int tm_min;    /* минути, 0-59 */  
    int tm_hour;   /* час, 0-23 */  
    int tm_mday;   /* ден от месеца, 1-31 */  
    int tm_mon;    /* месец от януари, 0-11 */  
    int tm_year;   /* година от 1900 */  
    int tm_wday;   /* ден от седм. от неделя, 0-6 */  
    int tm_yday;   /* дни от 1-ви януари, 0-365 */  
    int tm_isdst   /* лятно време */  
}
```

лятно време:

>0 – да в момента

0 – не в момента

<0 – няма информация

CLOCKS\_PER\_SEC – брой "цикли" в секунда,  
донякъде показва точността на отчитане  
clock\_t – брой цикли (long)

Получаване на текущата дата/час:

```
time_t now;
```

```
now = time(NULL);
```

ИЛИ

```
time_t now;  
time(&now);
```

Преобразуване между двата формата:

```
time_t now;  
time(&now);  
tm now_tm, nowgmt_tm;  
localtime_s(&now_tm, &now);  
cout<<now_tm.tm_mon;  
gmtime_s(&nowgmt_tm, &now);  
now = mktime(&now_tm);
```

Получаване/извеждане на стринг:

```
char s[30];  
time_t now;  
time(&now);  
ctime_s(s, 30, &now);  
cout << s;  
tm now_tm;  
localtime_s(&now_tm, &now);  
asctime_s(s, 30, &now_tm);  
cout << s;  
// Thu Jun 13 10:22:23 1991
```

strftime – преобразува в стринг с точно указан формат, справка – в help-a ;-)

Колко време работи програмата:

```
clock_t ct;  
ct=clock(); // връща брой "цикли", може да се  
            // раздели на CLOCKS_PER_SEC
```

Изтекло време (разлика)

```
time_t time1, time2;  
time1 = time(NULL);  
// някаква обработка  
time2 = time(NULL);  
double seconds = difftime(time2, time1);
```

// генерация на случайни числа

```
int number=rand();  
// случайно число между 0 и RAND_MAX  
случайно_до_100  
int n=rand()/(RAND_MAX/100);  
Предварителна инициализация:  
srand(число);  
напр.  
srand(time(NULL));
```