CODE ACADEMY

ОБУЧЕНИЕ ПО ПРОГРАМИРАНЕ

Среща 3 Типове данни и променливи.

CODE ACADEMY

Да си припомним

Какво научихме предния път?

CODE ACADEMY

Да прегледаме как се справихте със задачите за самоподготовка.

CODE ACADEMY

Променливи и тяхното съхранение в паметта

- Данните се записват в паметта и мястото им се определя от адрес.
- В езика С, локализирането на данните става и чрез име на променлива.
- Данните в програмата са стойности на променливи и те
  - могат да бъдат **прочетени на вход;**
  - могат да се **инициализират;**
  - могат да се **актуализират.**
- С оператора за присвояване на променливата се присвои стойност или израз.
- Всички типове данни се записват в паметта като се кодират чрез последователност от 0 и 1 с крайна дължина.

CODE ACADEMY

Съхранение на цели числа от тип int

- Типът **int** не включва цялото множество на целите числа, защото заема 4 байта.
- Типът **int** представя стойности на целите числа в диапазана от **-2** до **2** **– 1**
- В **32 бита** цяло число без знак може да се приеме стойности между **0** до **2** **– 1**
- Когато числото се представя със знак, един бит от двоичният му запис е знаков.
- В <limits.h> са дефинирани константи за min и max стойностите за всеки тип

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

int main() {
    printf("The minimum value of INT = %d\n", INT_MIN);
    printf("The maximum value of INT = %d\n", INT_MAX);

}
```

```
The minimum value of INT = -2147483648
The maximum value of INT = 2147483647
```

CODE ACADEMY

Модификатори на тип int

- **Signed –** цяло число със знак.
  Знакът може да не се записва, когато числото е положително;
- **Unsigned** - цяло число без знак. *Стойностите са **>0** или **0** ;*
- **Short** – с два пъти по-малко на брой байтове (int= 4 Bytes, short int= 2 Bytes);
- **Long** - с два пъти повече на брой байтове (int= 4 Bytes, long int= 8 Bytes);
  *Може да има вариации в размера на **long** и **long long**.*

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

int main() {
    printf("SHORT INT = %d Bytes\n", sizeof(short));
    printf("The minimum value of SHORT INT = %d\n", SHRT_MIN);
    printf("The maximum value of SHORT INT = %d\n", SHRT_MAX);
    printf("LONG INT = %d Bytes\n", sizeof(long));
    printf("The minimum value of LONG = %ld\n", LONG_MIN);
    printf("The maximum value of LONG = %ld\n", LONG_MAX);
}
```

```
SHORT INT = 2 Bytes
The minimum value of SHORT INT = -32768
The maximum value of SHORT INT = 32767
LONG INT = 8 Bytes
The minimum value of LONG = -9223372036854775808
The maximum value of LONG = 9223372036854775807
```

CODE ACADEMY

Типове за цели числа

тип / x64, little-endian	размер	диапазон
short int	2 Bytes (16 bits)	[-32 767, +32 767]
signed short	2 Bytes (16 bits)	[-32 767, +32 767]
unsigned short	2 Bytes (16 bits)	[0, 65 535]
int	4 Bytes (32 bits)	[−2 147 483 647, +2 147 483 647]
signed int	4 Bytes (32 bits)	[−2 147 483 647, +2 147 483 647]
unsigned int	4 Bytes (32 bits)	[0, 4 294 967 295]
signed long	4 Bytes (32 bits)	[-9223372036854775808 to 9223372036854775807]
unsigned long	4 Bytes (32 bits)	[0 to 18446744073709551615]
unsigned long long int	8 Bytes (64 bits)	[0, 18 446 744 073 709 551 615]
signed long long int	8 Bytes (64 bits)	[−9 223 372 036 854 775 807, +9 223 372 036,854,775,807]

Тип float - приближение на реалните числа

- Реалните числа могат да бъдат ирационални и да се представят с безброй много цифри, напр. *pi=3.141593*... Паметта на компютъра е ограничена и е не може да съхранява толкова много цифри.

- Тип **float** като приближение на реалните числа е ограничен в 4 байта.
- **float** се представят в **32 бита**, но **различно** представяне в **32 бита** от **int.**

- В типа **float** са дефинирани и специални стойности за обработка на грешки **NaN** – not a number и **INFINITY** - + безкрайност
- **min и max** стойности **за реалните числа** се дават чрез **макроси** от **<float.h>**

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

int main () {
   printf("The maximum value of float = %.10e\n", FLT_MAX);
   printf("The minimum value of float = %.10e\n", FLT_MIN);
}
```

```
The maximum value of float = 3.4028234664e+38
The minimum value of float = 1.1754943508e-38
```

Символен тип - char късо цяло число в 1 байт

Тип	Размер - CHAR_BITS	Диапазон
char	1 Byte	
unsigned char	1 Byte	[0, 255] 0 - 1111 1111(FF)
signed char	1 Byte	[-128,+127] 1000 0000 - 0111 1111
wchar_t	2 - 4 Byte	достатъчно голям за да побере и най-големия символ

char	decimal	binary	hex
0	48	0011 0000	30
9	57	0011 1001	39
A	65	0100 0001	41
Z	90	0101 1010	5a
a	97	0110 0001	61
z	122	0111 1010	7a

#include<wchar.h> библиотека от разширението на езика C

Binary и Hex

- Досадно е да се пише дълъг стринг от двоични цифри 0 и 1. Прието е за удобство да се използва 16-тичен запис с цифрите 0-9, A(10), B(11), C(12), D(13), E(14), F(15).
- За да се конвертира двоичен запис към 16-тично представяне, низа от 0 и 1 се групира в блокове от по 4 бита, като се започне от страната на единиците.
- Всеки блок от 4 бита се замества от съответната 16-тична цифра.

0011	1110	0101	1011	0001	1101	0110	1001
3	E	5	B	1	D	6	9

16-тичната константа записваме като **0x3E5B1D69 или 0x3e5b1d69**

Примери

- **1) 7529 = 0000 0001 1101 0110 1001 = Ox00001D69 = Ox1D69**

-7529 = 1111 1110 0010 1001 0111 = OxFFFFE297 = Ox1D69

2) 7529.0 =0 1000 1011 110 1011 0100 1000 0000 0000

-7529.0 =1 1000 1011 110 1011 0100 1000 0000 0000

Тези две представяния на реалното число са различни от представянето на целите числа.

3) ’A’ ≡ 0100 0001B ≡ 0x41

’1’ ≡ 0011 0001B ≡ 0x21.

Различно е от представянето на 1 и 1.0

Променливи и тяхното представяне в паметта

- Записването на променлива в паметта е свързано със заделяне на памет.
- Отделеното пространство е според типа дани и е с фиксиран размер за съхранение. За тип int този размер е 4 байта.
- В заделената памет са останали случайни данни (боклук), които са били изтрити при работа на други процеси.
- Адресът на заделената клетка памет се нарича l – value.
- По време на декларирането на променливата, в пространството от паметта, където ще се запише тази променлива може да се запише начална стойност. Процесът на задаване на начална стойност на променливата се нарича инициализилане.
- По време на работа на програмата, стойността на променливата може да се актуализира с нова стойност и това става с помощта на оператор за присвояване. Често се присвоява израз, в който участва старата стойност.

Променливи и оператор за присвояване

- **Записаната стойност в клетката памет, отделена за променлива се нарича r-value**

  Пример,

  int count=10;

  съдържанието 10 на променливата **count** е нейната **r-value**

- **Ако int count=10;**
  count=count*2+5;
  то r-value на променливата count ще бъде нейната финална стойност 25

- Адресът на клетката, в която се съхранява стойността на променливата **count,** т.е. **&count** е **l-value** на променливата **count**.

Променливи в паметта. Константи.

- Адресът или l – value на променливата може да бъде извлечен с помощта на унарния оператор &. Така адресът на променливата price ще бъде &price.

- Адресът на локацията може да бъде съхранен в друга променлива pointer или указател.

- Пример:
  int count =10, *cP; Указателят cP ще бъде от тип int *;
  cp=&count;

- Отделянето на памет за другите типове данни се отличава по размера на отделената памет, който зависи от типа данни.

- Константата се дефинира като променлива, но отпред е записано const
  Тя не може да си променя стойността, защото се записва в read-only памет.

- Пример: const double PI=3.1415926535897932

CODE ACADEMY

Въпроси?

Връзка между тип int и тип char

- **Ако данни от тип char се присвоят на променлива от тип int, в паметта отделена за целочислената променлива ще се съхрани ASCII кода на символните данни.**

- **Ако данни от тип int, които са 32 битови, се присвоят на променлива от тип char, която заема 8 бита, в локацията, предназначена за символната променлива ще се запазят 8-те най-значими бита от двоичното представяне на целочислената променлива.**

```
#include<stdio.h>

int main()
{
  int count ='C';
  char grade=1345;
  printf("count: %d, grade: %c\n",count, grade);
}
```

count: 67, grade: A

ASCII кода на 'C' е 67, а вътрешното представяне на 1345 е
0000 0000 0000 0000 0000 0101 0100 0001
Десетичната стойност на последните значими 8 бита е 65 -> 0100 0001,
което е ASCII кода на символа 'A'

Компютърна аритметика

- **Числото 1.3 е различно от стойността на променлива от тип float, инициализирана с 1.3**
  float a=1.3;

```
#include<stdio.h>

int main()
{
  float a=1.3;

  if(a==1.3)
          printf("Equal\n");
      else
          printf("Not equal\n");
}
```

Компютърна аритметика

- **Делението на целочислени данни int на нула дава грешка по време на изпълнение**

Делението на данни от тип float или double на нула не генерира никаква грешка по време на изпълнение, но резултатът е INFINITY - безкрайност

```
#include<stdio.h>
#include<math.h>
int main()
{
  int p=10, q;
  printf("Enter an integer: ");
  scanf("%d", &q);
  printf("p/q: %d\n", p/q );

  float n=10.0, m, r;
  printf(“Enter an integer: “);
  scanf(“%f”, &m);
  printf(“n/m: %f\n”, r= n/m );
  printf(“atan(%f)=%f\n”, r, atan(r));

 }
```

Компютърна аритметика

- Събирането може да даде грешен резултат поради препълване overflow

2147483647 + 1 = -2147483648

```
#include<stdio.h>
int main()
{
 int n=2147483647;
 printf("n+1 : %d\n", n+1);
}
```

- Може да има загуба на точност при аритметични действия с float

1.000000e-005 + 1.000000e+005 = 1.000000e+005

```
#include<stdio.h>
int main()
{
 float a=1.0e-5, b=1.0e+5, c;
  c=a+b;
 printf("%e + %e = %e\n", a, b, c);
}
```

Въвеждане – четене на данни от тип char

Програмата очаква да се въведат два символа

```
#include<stdio.h>
        int main()
        {
          char c,d;
          printf("Enter two characters: ");
          scanf("%c", &c);
          scanf("%c", &d);
          printf("%c..%c\n", c, d);
   }
```

Ако въведем символите като натиснем enter или space или tab, програмата няма да прочете вторият символ, защото като втори символ ще интерпретира интервала или новия ред.

Когато се четат два символа, трябва да се въведат един след друг

Ако искаме да прочетем символите, като всеки го напишем на нов ред, трябва да се оставят един или повече интервала (gap) в кавичките преди placeholder %c

```
scanf("    %c", &d);
```

PRE и POST INCREMENTS. PRE и POST DECREMENTS.

```
int count=10,   total =10;
1)
//++count;        → pre-increment
printf("%d” , ++count);

//total++; ;       → post-increment
printf("%d”,  total++);
```

И двата оператора увеличават стойността с единица, но при изпълнение на горния код, ще ке отпечата, че стойността на count е 11, докато отпечатаната стойност на total ще бъде 10.

```
2)
--count;        → pre-decrement
printf("%d” , count);

total--; ;       → post-decrement
printf("%d”,  total);
```

CODE ACADEMY

Упражнение

Упражнение 1:

Съберете unsigned char (255 + 10).

- Какво ще се получи?
- Защо?

```
printf("%d",(char)(250+10));
```

CODE ACADEMY

Отговор

```
int main(void){
    printf("%d", (unsigned char)(255 + 10));
    return 0;
}
```

CODE ACADEMY

Типове за числа с плаваща запетая

Тип	Размер	Диапазон
Float	4 Bytes (32 bits)	Приблизително ±3.40282347E+38F (6–7 значещи цифри след запетаята)
Double	8 Bytes (64 bits)	Приблизително ±1.79769313486231570E+308 (15 значещи цифри след запетаята)
Long Double	16 Bytes (128 bits)	(прецизност до 19-та цифра след запетаята)

Типове за числа с плаваща запетая

IEEE 754

float - четири байта (32 bits)

double - осем байта (64 bits)

long double - 16 байта (128 bits)

Според стандарта за представяне на числата с плаваща запетая, те се представят с три компонента:

{ S, E, M } -> $M * 2^{E}$, където

S е знак, E - порядък, M -мантиса

https://en.wikipedia.org/wiki/IEEE_754

CODE ACADEMY

ОТ КАКЪВ ТИП ЩЕ ДЕФИНИРАТЕ:

- Числото ПИ.
- Броя на етажите в една сграда.
- Цена за кутията с шоколад.
- А за кутията с бонбони?
- Ден от седмицата.
- Дали спя или не.
- Какъв е шанса да спечеля от тотото?
- А какъв е шанса да завали?
- Вашето име
- Отговорът на въпроса “Обичате ли да програмирате?”
- Броят на мравките в мравуняка.
- А в Европа?
- Рестото, което няма да ви върнат?
- Размерът на обувките ви?
- След колко време ще дойде автобусът?
- Номерът на входа ви?

CODE ACADEMY

bool тип

Тип bool в C няма. Всяко ненулево число се възприема като true, а всяко число със стойност 0 се възприема като false.

Състоянието на true е 1, а false е 0.

printf няма формат спецификатори за типа bool и затова използва този на типа int - %d

CODE ACADEMY

ЛОГИЧЕСКИ ОПЕРАЦИИ

1. **Логическа променлива -** променлива, която приема само две стойности **true**(истина) и **false**(лъжа).

2. **Операции(действия) с логически променливи**

а) **Логическо събиране - ИЛИ ||**
a||b означава, че поне едно от двете е вярно

true || true <==> true
true || false <==> true
false || true <==> true
false || false <==> false

ИЗВОД **a||b** е **false** само когато и двете са **false.**
Във всички останали случаи е **true.**

CODE ACADEMY

ЛОГИЧЕСКИ ОПЕРАЦИИ

b) Логическо умножение - И &&

a && b означава, че и двете едновременно са вярни.

true && true <==> true
true && false <==> false
false && true <==> false
false && false <==> false

ИЗВОД **a && b** е **true** само когато и двете са **true.**
Във всички останали случаи е **false.**

c) Отрицание - НЕ !

!true <==> false
!false <==> true

!(a<b) <==> b<=a
!(a==b) <==> a!=b
!(a>=b) <==> a<b

CODE ACADEMY

ПРАВИЛА - закони на Бул

1) ДВОЙНО ОТРИЦАНИЕ

!(! a) <==>a

!(!true) <==> true
!(!(a<b)) <==> !(a>=b) <==> a<b

2) ОТРИЦАНИЕ НА ИЛИ

!(a||b) <==> (!a) && (!b)

!(a<2 || a>6) <==> !(a<2) && !(a>6) <==> (a>=2) && (a<=6)

3) ОТРИЦАНИЕ НА И

!(a && b) <==> (!a)||(!b)

CODE ACADEMY

Въпроси?

CODE ACADEMY

ПРАВИЛА - закони на Бул

1) ДВОЙНО ОТРИЦАНИЕ

!(! a) <==>a

!(!true) <==> true
!(!(a<b)) <==> !(a>=b) <==> a<b

2) ОТРИЦАНИЕ НА ИЛИ

!(a||b) <==> (!a) && (!b)

!(a<2 || a>6) <==> !(a<2) && !(a>6) <==> (a>=2) && (a<=6)

3) ОТРИЦАНИЕ НА И

!(a && b) <==> (!a)||(!b)

CODE ACADEMY

ПРАВИЛА - закони на Бул

1) ДВОЙНО ОТРИЦАНИЕ

!(! a) <==>a

!(!true) <==> true
!(!(a<b)) <==> !(a>=b) <==> a<b

2) ОТРИЦАНИЕ НА ИЛИ

!(a||b) <==> (!a) && (!b)

!(a<2 || a>6) <==> !(a<2) && !(a>6) <==> (a>=2) && (a<=6)

3) ОТРИЦАНИЕ НА И

!(a && b) <==> (!a)||(!b)

CODE ACADEMY

Въпроси?

CODE ACADEMY

Задачи

6. Дефинирайте променливи със стойност -127, 255, 63 498, 4 294 967 292, -9 000 000 000 000 775 845.
Изведете всяка променлива на екрана със printf()

7. Дефинирайте променливи със стойност 24 212, -1 357 674, 1 357 674, -1 357 674 000, 3 657 895 000.
Изведете всяка променлива на екрана със printf()

CODE ACADEMY

Задачи

8. Да прочетем какво се случва със стойността на променливата x:

```
char x = 8;
x = x -2;
x= x+ 6;
x= x – 10 + 2;
```

Каква ще бъде стойността на x, ако я отпечатаме на екрана след последната калкулация?

CODE ACADEMY

Задачи

9. Дефинирайте променливи със стойност

4.9876543, 7.123456789012345678890, 18 398 458 438 583 853.28, 18 398 458 438 583 853.3987593728492842 2.

Изведете всяка променлива на екрана с printf()

CODE ACADEMY

Въпроси?

CODE ACADEMY

КАКВИ БРОЙНИ СИСТЕМИ ИМА?

- Десетична
- Шестнадесетична `0x`
- Осмична `0`
- Двоична `0b`

CODE ACADEMY

Бройни системи

DECIMAL	BINARY	OCTAL	HEXADECIMAL
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
…	…	…	…
9	1001	011	9
10	1010	012	A
11	1011	013	B
12	1100	014	C
13	1101	015	D
14	1110	016	E
15	1111	017	F
…	…	…	…
99	1100011	143	63
100	1100100	144	64

CODE ACADEMY

ТЕЖЕСТ НА ВСЯКА ЦИФРА В ПОЗИЦИОННА БРОЙНА СИСТЕМА

Позиция – при позиционните бройни системи стойността на числото нараства в зависимост от позицията му дясно от на ляво. Базата, с която нараства, зависи от бройната система.

Например:

при десетично цяло число $1677 = 1*10^3 + 6*10^2 + 7*10^1 + 7*10^0$

при шеснадесетично цяло число $0x1677 = 1*16^3 + 6*16^2 + 7*16^1 + 7*16^0$

$4096 + 1536 + 112 + 7*1 = 5751$

Същите принципи важат при двоичната и осмичната бройни системи, като се взимат степени на 2 и 8 съответнно.

CODE ACADEMY

ПРЕОБРАЗУВАНЕ ОТ ДЕСЕТИЧНО В ДВОИЧНО ЧИСЛО

число	делено на 2	остатък
29	14.5	1
14	7	0
7	3.5	1
3	1.5	1
1	0.5	1

Двоично число 11101 – записваме стойностите в посока от последно получено нагоре.

Как ще стане в осмична бройна система?

Как ще стане в шеснадесетична бройна система?

ПРЕОБРАЗУВАНЕ ОТ ДВОИЧНА В ДЕСЕТИЧНА СИСТЕМА

Двоично число

0 0 1 1 1 0 1

*

2^6 2^5 2^4 2^3 2^2 2^1 2^0

$0*2^6 + 0*2^5 + 1*2^4 + 1*2^3 + 1*2^2 + 0*2^1 + 1*2^0 =$

$= 0 + 0 + 16 + 8 + 4 + 0 + 1 =$

$= 29$

Десетично число

Съкратени преобразувания

I. Преобразуване от двоична в шестнадесетична система.

Правило: **Групират се битовете в групи по четири. Всяка група от четири бита се конвертира в шестнадесетична цифра. При необходимост се добавят водещи и/или крайни нули.**

Преобразуване на 0101 1111 1011 в шестнадесетична бройна система:

0000 0101 1111 1011

0 5 F B

0000 0101 1111 1011 = 0x05FB

II. Преобразуване от шестнадесетична в двоична система

Правило: **Всяка шестнадесетична цифра се представя чрез четири двоични цифри. Ако е необходимо, се добавят водещи нули.**

Преобразуване на 47C,A6 (16) в двоична бройна система:

0 4 7 C
0000 0100 0111 1100
0x047C = 010001111100

III. Преобразуване от двоична в осмична система.

Правило: **Групират се битовете в групи по три, започвайки от десетичната запетая надясно и наляво. Всяка група от три бита се конвертира в осмична цифра. При необходимост се добавят водещи и/или крайни нули.**

Преобразуване на 10111111011 в осмична бройна система:

010 111 111 011

2 7 7 3

10111111011 = 2773

IV. Преобразуване от осмична в двоична система.

Правило: **Всяка осмична цифра се представя чрез три двоични цифри. Ако е необходимо, се добавят водещи нули.**

Преобразуване на 573(8) в двоична бройна система:

5	7	3
101	111	011

573 = 101111011

CODE ACADEMY

V. Преобразуване от шестнадесетична в осмична система.

Правило 8: Използват се две съкратени преобразувания:
1) от шестнадесетична в двоична
система;
2) от двоична в осмична система.

VI. Преобразуване от осмична в шестнадесетична система

Правило 9: Използват се две съкратени преобразувания:
1) от осмична в двоична система;
2) от двоична в шестнадесетична система.

CODE ACADEMY

23 --> ?

23:2 =11 + (1)
11:2 =5 + (1)
5:2 =2+ (1)
2:2=1+ (0)
1:2=0+ (1)

23 = 10111

10111 =1+1*2+1*4+0*8+1*16=1+2+4+16=23

CODE ACADEMY

A3F1 **16 --> ?**

A 3 F 1

1010 0011 1111 0001

001 010 001 111 110 001

1 2 1 7 6 1

==========

A3F1 = 121761

4A

=A+4*16 =10+64=74

CODE ACADEMY

23 --> ?

23:8=2 +(7)
2:8 =0 +(2)

23 = 27

000 0
001 1
010 2
011 3
100 4
101 5
110 6
111 7

CODE ACADEMY

Въпроси?

Оператори

Операторите се прилагат върху променливи от даден тип (операнди) и се получава резултат от тип, който зависи от типа на операндите

Според броя на аргументите, операторите биват:

. **унарен:** <op> A, B <op>

```
&uValue
Uvalue--
```

. **бинарен:** A <op> B

```
1 + 2
uValue & 0xABCD
```

. **тернарен** (3 аргумента)

```
(uA > uB) ? uA : uB
```

CODE ACADEMY

Класификация на оператори

- аритметични +; -, *, /, %
- релационни: 1 > 2, <, <=, >=
- логически &&, ||,
- побитови &, |, ^, ~
- оператори за присвояване: =, +=, -=, *=, /=
- условен оператор: ? а : б
- оператор за изброяване: запетая ,
- други оператори: вземане на адрес &, вземане на стойност *

CODE ACADEMY

Приоритет на оператори

OPERATOR	TYPE	ASSOCIAVITY
( ) [ ] . ->		left-to-right
++ -- + - ! ~ (*type*) * & sizeof	Unary Operator	right-to-left
* / %	Arithmetic Operator	left-to-right
+ -	Arithmetic Operator	left-to-right
<< >>	Shift Operator	left-to-right
< <= > >=	Relational Operator	left-to-right
== !=	Relational Operator	left-to-right
&	Bitwise AND Operator	left-to-right
^	Bitwise EX-OR Operator	left-to-right
|	Bitwise OR Operator	left-to-right
&&	Logical AND Operator	left-to-right
||	Logical OR Operator	left-to-right
? :	Ternary Conditional Operator	right-to-left
= += -= *= /= %= &= ^= |= <<= >>=	Assignment Operator	right-to-left
,	Comma	left-to-right

CODE ACADEMY

Приоритет на оператори

```
#include <stdio.h>
int main(void){
int x = 1;
int y = 2;
char t = 0, f = 0;
t = x < y;
f = x > y;
printf("True = %d\n", y > x);
printf("True = %d\n", t);
printf("False = %d\n", f);
y = 0;
f = x && y;
printf("False && %d\n", f);
t = x || y;
printf("t || %d\n", t);

y = x + 2 * 3;
printf("y = %d\n", y);

y = (x + 2) * 3;
printf("y = %d\n", y);
return 0;
}
```

Приоритет на оператори

Задача 3: Имаме две променливи int x = 10; и int y = 10; създайте две променливи int t, f; На първата присвоете резултата от сравнението на **x == y**, а на втората **x != y** принтирайте с printf променливите t и f.

Задача 4. Какъв ще бъде резултатът в променливата t:

```
      int x = 10;
int y = 10;
int z = 30;
int t = x==y + (z < y) != 20;
```

Приоритет на оператори

Задача 5:

a) В какъв ред ще се изпълнят операциите?

```
t = x && y & x << 1;
```

Какво ще се принтира на екрана?

b) В какъв ред ще се изпълнят операциите?

```
t = x && y^x <<1;
```

Какво ще се принтира на екрана?

Приоритет на оператори

Запетайката поставена в скоби (a, b, c) е последователност от изрази, разделени със запетаи, която изчислява до последния израз c, докато

```
{
    a, b, c;
}
```

е поредица от изрази и не изчислява никаква стойност.

CODE ACADEMY

Приоритет на оператори

Задача 7:

В какъв ред ще се изпълнят оператора запетая , ?

```
#include <stdio.h>

int main() {
	int x = 1, y = 0;
int z = y || x;

	printf("\nInit values: X= %d, Y= %d\n", x, y);
	x = 1 + 2, 2 * 3, 3 + 4;
	y = ( 7 * 8, 8 + 9, 9 - 4);
	printf("\nX= %d, Y= %d\n", x, y);
	return 0;
}
```

Приоритет на оператори

Задача 8: Инкрементиране и декрементиране префиксно и постфиксно:

```
#include <stdio.h>

int main() {
int i = 0, j = 0;
      printf("i = %d, j = %d\n", i, j);
      printf("j=i++: %d\n", j=i++);
      printf("i = %d, j = %d\n", i, j);
      printf("j = ++i: %d\n", j=++i);
      printf("i = %d, j = %d\n", i, j);
      printf("i--: %d\n", i--);
      printf("i = %d, j = %d\n", i, j);
      return 0;
}
```

Приоритет на оператори

Задача 9: Условен оператор (?:)

```
#include <stdio.h>

int main()

{

  int nA = 1;

  int nB = (nA == 1) ? 2 : 0; /* сравнява число */

  printf("A value is %d\n", nA); printf("B value is %d\n", nB);

}
```

a) въведете стойност за nA с scanf

b) използвайте условния оператор за намиране на максималното от две числа

CODE ACADEMY

Задачи

Упражнение . Increment / decrement operators

```
#include <stdio.h>
int main() {
  int iCounter = 0;
  while( ++ iCounter < 3 ) {
    printf("Counter %d\n", iCounter);
    printf("++ Counter: %d\n", ++ iCounter);
    printf("-- Counter: %d\n", -- iCounter);
    printf("Counter ++: %d\n", iCounter ++);
    printf("Counter --: %d\n", iCounter --);
  }
  return 0;
}
```

CODE ACADEMY

Задачи

Упражнение . Relational operators

```
#include <stdio.h>
int main() {
  int nX = 33; /* homework, input with scanf */
  int nY = 20;
  if (nX == nY) {
    printf("%d and %d are equal\n", nX, nY);
  } else {
    printf("%d and %d are not equal\n", nX, nY);
  }
  if (nX > nY) { printf("%d is greater than %d\n", nX, nY); }
} /* използвайте и останалите оператори за сравнение */
```

Задачи Логически оператори

```
#include <stdio.h>
 int main() {
 int nA = 40; /* използвайте scanf */
 int nB = 20;
 int nX = 20;
 int nY = 30;
 if ( nA > nB && nA !=0 ) {
    printf("&& Operator : Both conditions are true\n");
 }
 if ( nX > nY || nY != 20) {
    printf("|| Operator : Only one condition is true\n");
 }
 if ( ! (nA > nB && nB !=0 ) ) {
    printf(" ! Operator : Both conditions are true\n");
 } else {
    printf("Both conditions are true.\n");
 }
}
/* опитайте различни комбинации*/
```

Задачи

Упражнение Оператор за вземане на адрес (&) и дереференция (*)

```
#include <stdio.h>
int main()
{
   int iA = 13;
   int* pValue = NULL; /* pointer to int */
   pValue = &iA; /* assigning address of iA to the pointer */
   printf("\nAddress of variable iA is: %p\n", pValue);
   printf("\nValue of variable iA is: %d\n", *pValue);
   return 0;
}
```

CODE ACADEMY

Задачи – упражнение

Задача 1 Напишете програма, която намира лицето на правоъгълник със страни равни на първата и последната цифра на въведено трицифрено число. Програмата трябва да отчете, че някои от цифрите могат да бъдат равни на нула.

Задача 2 Да се напише програма, която чете едно цяло трицифрено число и извежда броя на еднаквите цифри в него.

CODE ACADEMY

Въпроси?

CODE ACADEMY

Задачи – домашна работа

Задача 1 Да се напише програма, която чете едно цяло шестцифрено число, отделя от него цифрите на позиции 1,3,5 и 6 и с тези цифри образува ново число от четири цифри. Ако разглеждаме това ново число като година,програмата трябва да изведе дали тази година е високосна или не.

Задача 2 Да се напише програма, която чете едно цяло шестцифрено число и намира лицето на правоъгълника със страна a, равна на сумата от не-четните цифри и страна b, равна на сумата от четните цифри.

Задачи – домашна работа

Задача 3 Представете си, че имате фирма за отдаване на каравани и кемпери под наем, за която трябва да разработите софтуер. Вие сте малка фирма и имате общо 3 каравани на цена 90 лв. на ден и 3 кемпера на цена 100 лв. на ден. Декларирайте променливи за броя на караваните, цената на караваните, броя на кемперите и цената им. Направете меню, което пита клиента каравана или кемпер ще иска. Направете променлива, в която да се събират парите, които клиента дължи на компанията. Принтирайте резултата.

Задача 4 Създайте С програма, която калкулира стойността на покупки в магазин за хранителни стоки. Клиентът може да закупи различна комбинация от продукти. В магазина има следните налични продукти и цени:

Домати- 4.5 лв.за килограм, Брашно 1.80 лв. за килограм,

Кисело мляко - 0.50 лв. за брой, Сладолед на фунийки 0.60 лв. броя

Бонбони 1.50 лв. за килограм, Близалки 0.15 лв. за брой.

Задачи – домашна работа

Задача 5 Направете бягаща светлина, като приемете, че всеки бит от дадена променлива, е свързан с лампа (или светодиод). Когато битът е нула, лампата не свети, когато е единица свети.

Примерно, ако генерирате последователност:

1, 2, 3, 4, се получава следното:

```
1 *...................
2 .*..................
4 ..*.................
8 ...*.................. ...
```

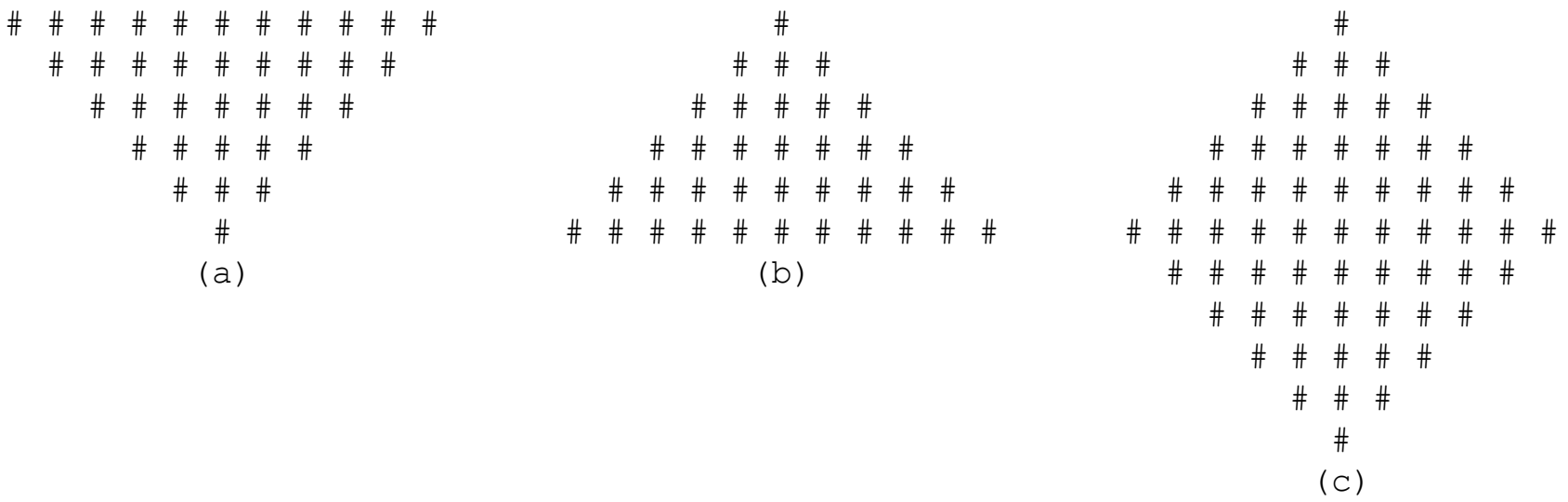
За да генерирате закъснение, използвайте следната функция от C runtime:

```
#include <unistd.h>
unsigned int sleep(unsigned int seconds);
```

Задачи – домашна работа

6. Напишете програма, която изчислява броя на секундите в една година.

7. Напишете програма PrintPatterns, която отпечатата следните текстови графики на екрана. Графиките една под друга:



CODE ACADEMY

Упражнение Домашно

Оператори за присвояване

```
# include <stdio.h>
int main() {
  int nResult = 13; /* опитайте с други стойности */
  int nX = 4;
  printf("Result = %d\n", nResult);
  nResult += nX;
  printf("Result += %d -> %d\n", nX, nResult);
  nResult = 1;
  nResult <<= nX;
  printf("Result <<= %d -> %d\n", nX, nResult);
} /* използвайте и други оператори за присвояване */
```

Упражнение Домашно

Аритметични оператори

```
#include <stdio.h>
int main()
{
    int iX = 13;
    int iY = 44;
    int iResult = 0;
     iResult = iX - iY;
    printf("%d - %d = %d \n", iX, iY, iResult);
    iResult = iY / iX;
    printf("%d / %d = %d \n", iY, iX, iResult);
    iResult = iY % iX;
    printf("%d / %d Remainder: %d\n", iY, iX, iResult);
    /* homework: examples for '*' и '+', float - използвайте вместо int */
    return 0;
}
```

CODE ACADEMY

Упражнение Домашно

Направете генератор на случайни числа по следния алгоритъм (xorshift):

```
int A = 1, акумулатор с начална стойност
При всяко вземане на число от генератора се прави следното:
   (i) A ^= A << 13;
   (ii) A ^= A >> 17;
   (iii) A ^= A << 5;
```

При всяка итерация изведете числото на екрана.
(Направете вариация с акумулатор long long int)

CODE ACADEMY

Упражнение Домашно

1. Да се преобразуват от десетична в двоична бройна система дадените числа, като се използват алгоритмите за преобразуване:

а) 23 б) 128 в) 34 г) 256

2. Да се преобразуват от десетична в осмична бройна система числата:

а) 245 б) 64 в) 128 г) 166

3. Да се преобразуват от десетична в шестнадесетична бройна система числата:

а) 312 б) 256 в) 123 г) 317

CODE ACADEMY

Упражнение Домашно

4. Да се преобразуват от двоична в десетична бройна система дадените числа, като се използва формулата за представяне:

а) 100011 б) 1000000 в) 1101001 г) 1001

5. Да се преобразуват от шестнадесетична в десетична бройна система числата:

а) 16F б) 19 в) 4A г) 7F

6. Да се преобразуват от осмична в десетична бройна система числата:

а) 53 б) 100 в) 151 г) 23

CODE ACADEMY

Упражнение Домашно

Като се използват съкратени преобразувания, да се извършат съответните действия:

7) 1000111001 (2) – в осмична бройна система;

8) 1100111010 (2) – в шестнадесетична бройна система;

9) 152 (8) – в двоична бройна система;

10) 5A6C (16) – в двоична бройна система;

11) 53 (8) – в шестнадесетична бройна система.

CODE ACADEMY

КАКВО НАУЧИХМЕ ДНЕС?