Съставяне, въвеждане и настройка на програми, илюстриращи операциите в езика и основните типове данни

Структура на програма

<библиотеки>

…деклариране на глобални променливи и спомагателни функции

<главна функция>

{

< тяло на функцията>

}

<библиотеки> - в тях се съдържат основни и спомагателни методи, оператори и функции по подразбиране. Библиотеките се изброяват една под друга, като започват **с #include** след което се записва пълен път до файла, който искаме да използваме за библиотека, а ако е от подразбиращите се стандартизирани в езика само името му. Обикновено библиотеките завършват с **.h**, което идва от така наречените **header** файлове – спомагателни с готов за ползване код. Header файлове може да направите и сами, като за използването им е нужно да спазите гореописаните правила.

Когато е от стандартните библиотеки то извикване по следния начин:

#include <име на библиотека .h>

Когато е създадена от нас:

#include “пълен път и име на файла, който ще използваме”

Между библиотеката и главната функция могат да бъдат декларирани и описвани други функции, които играят спомагателна роля за работата на програмата.

Променливите описани извън функции в основното тяло на програмата се наричат ***глобални***. За тях ще стане дума в следващата точка от упражнението.

<главна функция> - главната функция – **main** – е основната функция, от която започва работата на програмата. Тя е задължителен елемент от програмата и може да съществува само една. Всяка друга функция играе роля на спомагателна към основната и развива изграждането на програмния код.

<тяло> на функция или оператор е мястото, което се огражда с **{ }**, между които пишем код с определена функционалност. Цялото съдържание на кода между скобите се изпълнява при едно логическо използване на оператора или функцията, към които е прикачен. Възможно е да имаме множество вграждания на <тяло> в <тяло>, като се спазва правилото, че скобата за затваряне - } – затваря последното отворено място – {. За да функционира правилно програмата трябва всички тела да са правилно отворени и затворени.

Пример за правилно написана програма:

```
#include <stdio.h>
 int main()
{
 printf("Hello World!\n");
return 0;
}
```

Понятие за променлива

Променливата в програмните езици има за цел да съхранява данни. За да се използва променлива в програмният език С, са необходими две неща: да се избере подходящ тип на данните – това е видът данни, който ще се съхранява в тази променлива, както и име на самата променлива. Записването на този избор в програмния код се нарича дефиниция. Всяка променлива, използвана в С, задължително трябва да бъде дефинирана. По този начин, компилаторът ще знае колко памет да задели и как да интерпретира тези данни.

Когато на променлива даваме стойност то тогава декларираме самата променлива като тя приема в паметта съответната стойност.

Видимост на променлива – в езика С видимост на променлива или живот означава мястото, на което може да използваме вече декларирана или дефиниране променлива.

Тук се влияем от няколко правила:

-Всяка променлива е видима след мястото си на дефиниране/деклариране, съответно може да бъде ползвана от там нататък в кода.

-Глобалните променливи се дефинират извън функциите в общата част на програмата, имат стойност 0 при дефиниция и могат да бъдат ползвани навсякъде в програмата след дефиницията им.

-Локалните променливи се използват само в блока код, в който са обявени и нямат стойност по подразбиране по време на дефиниция. Техният живот е в рамките на обявеното тяло. Локалните променливи са с по-голям приоритет при ползване в кода от глобалните.

Не могат да съществуват променливи с едни и същи имена и от един и същи вид в едно тяло!

Видове променливи в C

В програмния език C са създадени типове данни, които служат за изграждането на основата за съхранението на данните. Основните типове данни имат за цел да описват цели числа, реални числа и букви. Всеки един тип от тези данни се характеризира със следните параметри – размер байтове, които заема в паметта; интервал на стойностите, които могат да бъдат записвани чрез този тип.

Целочислени типове данни

Тип	Големина	Допустими стойнсти
char	8 бита	от -128 до 127
short	поне 16 бита	(16 бита) от -32 678 до 32 767
int	поне 16 бита	(32 бита) от - 2 147 483 648 до 2 147 483 647
long	поне 32 бита	(32 бита) от - 2 147 483 648 до 2 147 483 647
long long	64 бита	от -9 223 372 036 854 775 808 до 9 223 372 036 854 775 807

В таблицата са показани допустимите стойности за всеки един тип за компилатор GCC, разликите се в допустимите стойности се наблюдават, защото стандартът за езика C е гъвкав и има възможност да варират стойностите за различните компилатори.

Типове от данни описващи реални числа (числа с плаваща запетая)

Тип	Големина	Допустими стойнсти
float	(32 бита)	от +/- 1.17549e-038 до +/-3.40282e+038
double	(64 бита)	от +/- 2.22507e-308 до +/- 1.79769e+308
long double	(92 бита)	от +/- 3.3621e-4932 до +/- 1.18973e+4932

От таблицата се вижда, че числата в плаваща запетая като допустими стойности се различават драстично. Освен това типове като double и long double се препоръчва да се използват, когато точността на реалните числа е от значение.

Типовете данни, които описват реалните числа се наричат „числа с плаваща запетая“, защото в представянето на числата в компютъра се извършва използвайки записване чрез мантиса и порядък, които се записват на различни места в двоичния запис на числото

Типове от данни описващи букви

В програмният език С съществува един единствен тип данни, който описва букви и символи. Този тип е char (съкратено от character от англ. „буква, знак“). Този тип както беше споменато е и тип, описващ целочислени числа. Причината типът да има така дуалност (двойнствен характер) се крие в представянето на символите в една компютърна система. Всички възможно символи са записани в една таблица (ASCII таблица), в която срещу всеки символ е записано съответното му число. Така, когато трябва стойността на променливата от тип char да бъде представена като буква, компютърът автоматично заменя числовата ѝ стойност, със символа записан срещу нея. По този начин символите като ‘A’ имат стойност 65, а ако към буквата прибавим число получим символ, който седи на отстояние толкова места, колкото е стойността на числото.

Безнакови типове данни

В програмният език С има типове, които описват само цели положителни числа. Тези типове данни се получават като пред целочислените типове данни се запише unsigned. По този начин новите типове имат следните допустими стойности:

Тип	Големина	Допустими стойнсти
unsigned char	8 бита	от 0 до 255
unsigned short	поне 16 бита	(16 бита) от 0 до 65 535
unsigned int	поне 16 бита	(32 бита) от 0 до 4 294 967 295
unsigned long	поне 32 бита	(32 бита) от 0 до 4 294 967 295
unsigned long long	64 бита	от 0 до 18 446 744 073 709 551 615

Създаване и основни операции с променливи

Създаване на променливи Дефинирането на променливи се извършва в началото на тялото на всяка функция, според стандарта на C. За да бъде създадена една променлива е нужно да се следва един от следните два модела:

```
<тип> <име на променлива>;                          --> int k;

<тип> <име на променлива> = <начална стойност1>; --> double pi = 3.14;
```

В двата модела се заместват полетата както следва:
<тип> - това поле се замества с желаният тип, който е нужен за съхраняване на данните;
<име на променлива> - това е името на данните, което ще се използва за достъп то тези данни;
<начална стойност> - може да се зададе начална стойност на дадената променлива.

Важно е да се отбележи, че в програмният език C компилаторът не се грижи за зануляване на стойностите при тяхното дефиниране, за това е добра практика да се задава начална стойност. Този проблем може да се демонстрира и със следната логическа задача:

На Иванчо му дали 5 ябълки. След това той изял три. Колко ябълки му останали?

Отговор: Не се знае колко ябълки е имал преди да му бъдат дадени тези ябълки.

На един ред може да има дефинирана повече от една променлива от един и същи тип. Имената на променливите се разделят със запетая.

Задача: Съставете програма на C, като използвате дефиниция и декларация. Пробвайте различните типове данни.
Задача: Изведете цифрите от ASCII таблицата, направете същото и за малки и големи букви по избор.

Операции с елементарен тип данни

Операциите с елементарните типове данни се разделят на няколко типа - аритметични, логически и побитови (последният вид само за целочислени променливи).

Аритметични операции

- Присвояване – операция, чрез която се променя стойността на дадена променлива

```
int i = 7;
int k = 5;

i = k; // Новата стойност на i става стойността, на която е равна k, следователно
и ще бъде равно на 5
```

- Събиране – оператор +

```
int i = 7;
int k = 5;


i = k + 5; // новата стойност на i сумата от стойността на k и 5, следователно
новата стойност на и ще е 5+5=10

i = i + 3; // от дясната страна на израза i представлява старата си стойност,
новата стойност на i се получава след извършване на аритеметичната операция
```

- Изваждане – оператор –

```
int i = 7;
int k = 5;

k = i - 5;
k = 3 - k;
```

- Умножение – оператор *

```
int a = 5, b = 6;
int area;

area = a*b; // Стойността на area ще е равна на произведението на a и b
```

- Деление – оператор /

```
float a = 5;
float b;

b = a/2.5;
```

Важно е да се знае, че когато в делението участват цели числа (числител и знаменател) резултатът от делението ще е цяло число. Това означава, че ако делим 5 на 2, то резултатът ще е 2

- Остатък от деление – оператор % - операцията е възможна само за цели числа

```
char s = 7;
char p;

p = s % 3; // тук новата стойност на p ще е 1, защото при целочислено деление 7 на
3 резултатът е 2, а 2*3 = 6, следователно остатък 1
```

Задачи:

1. Да се напише програма, която намира лице на правоъгълник по зададени от клавиатурата две страни и го извежда на екрана.

2. Да се напише програма, която намира обиколката на окръжност по зададен диаметър от клавиатурата и извежда резултата на екрана.

Побитови операции в C

Програмният език C позволява работа с оператори за побитови операции: **NOT, AND, OR, XOR, SHIFT LEFT, SHIFT RIGHT**

Действията се извършват на ниво битове в двоичния запис на числото.

Побитово НЕ – NOT

- **Оператор в С:** ~
- **Брой променливи:** една
- **Действие:** сменя алтернативно стойностите на битовете в една променлива
- **Таблица на истинност:** показва действието за едни бит

x	~x
0	1
1	0

Побитово И – AND

- **Оператор в С:** &
- **Брой променливи:** две
- **Действие:** Резултатът от изпълнението е **1** в дадения бит, когато и двете промеливи в този бит имат стойност **1**,в противен случай е **0**
- **Таблица на истинност:** показва действието за едни бит

a	b	a & b
0	0	0
0	1	0
1	0	0
1	1	1

Побитово ИЛИ – OR

- **Оператор в С:** |
- **Брой променливи:** две
- **Действие:** Резултатът от изпълнението е **1** в дадения бит, когато поне една от двете промеливи в този бит имат стойност **1**, в противен случай е **0**
- **Таблица на истинност:** показва действието за едни бит

a	b	a | b
0	0	0
0	1	1
1	0	1
1	1	1

Побитово ИЗКЛЮЧВАЩО ИЛИ – XOR

- **Оператор в С:** ^
- **Брой променливи:** две
- **Действие:** Резултатът от изпълнението е **1** в дадения бит, когато поне една от двете промеливи в този бит имат различни стойност, ако стойностите са равно е **0**
- **Таблица на истинност:** показва действието за едни бит

a	b	a ^ b
0	0	0
0	1	1
1	0	1
1	1	0

Побитово ИЗМЕСТВАНЕ ВЛЯВО – SHIFT LEFT

- **Оператор в С:** <<
- **Брой променливи:** две
- **Действие:** Битовете на първата променлива се изместват на ляво с толкова места, колкото е стойността на втората променлива. Празните места се запълват с нули.
- **Пример:**

```
char a = 0xD3;
char y;

y = a << 3;  // a = 11010011(2)
             // y = 10011000(2) => y = 0x98
```

Побитово ИЗМЕСТВАНЕ ВДЯСНО – SHIFT RIGHT

- **Оператор в С:** >>
- **Брой променливи:** две
- **Действие:** Битовете на първата променлива се изместват на дясно с толкова места, колкото е стойността на втората променлива. Празните неща се запълват с нули
- **Пример:**

```
char a = 0xC5;
char y;

y = a >> 2;  // a = 11000101(2)
             // y = 00110001(2) => y = 0x31
```

Логически оператори в езика С

Логическите операции са част от операциите, с които може да се работи с елементарните типове данни.

В програмният език С под „истина“ се разбира ненулева стойност, а под „лъжа“ – нула.

Видове логически операции

- **Логическо НЕ „!“** – за разлика от побитовото НЕ където се прави преобразуване на ниво битове, логическото НЕ променя истинността на дадена променлива т.е. ако стойността е „лъжа“, то НЕ „лъжа“ е истина, и обратно.
- **Логическо И „&&”** – за да е „истина“ едно условие, трябва лявата и дясната страна на оператора да са „истина“
- **Логическо ИЛИ „||”** – за да е „истина“ едно условие, трябва да поне едно от двете стойности от ляво или дясно да е „истина“

Сравнения

Оператор	Значение
<	По-малко
>	По-голямо
>=	По-малко или равно
<=	По-голямо или равно
!=	Различно (не равно)
==	Равно

Забележка: *Да се внимава при употребата на сравняването за равенство, защото често се обърква с операторът за присвояване „=”. Резултатите от тази грешка могат да бъдат непредсказуеми.*

Примери

- (x > 5) && (x<=10)
- (x < 4) || (x >12)
- (x > 4) && (x!= 6)