УНИВЕРСИТЕТ ИМЕНИ СУЛЕЙМАНА ДЕМИРЕЛЯ

Жапаров М. К.

Microlearning in Programming Language C++

Учебно-методическое пособие

Каскелен, 2015

УДК 004. 4(075.8)

ББК 32.973.26-018 я73

Z 11

Рекомендовано к изданию Учебно-методическим советом Университета имени Сулеймана Демиреля протокол №1 от 25.08.2015 г.

Рецензенты:

Кожахметов К., PhD, ассистент профессора, Казахстанско-Британский технический университет

Амиргалиев Е. Н. д.т.н., профессор, Университет имени Сулеймана Демиреля

Zhaparov M.K.

Z11 Microlearning in Programming Language C++: Methodical-educational manual. Zhaparov M.K. — Kaskelen, Suleyman Demirel University, 2015. — 89 p. — In English.

ISBN 978-601-7537-23-4

Учебно-методическое пособие «Microlearning in Programming Language C++:» предназначена для бакалавров по специальностям 5В03000 — «Информационные Системы» и 5В040000 — «Вычислительная Техника и Программное Обеспечение».

Methodical-educational manual « Microlearning in Programming Language C++» is designed for bachelors of specialties 5B030000 — “Information Systems” and 5B040000 — “Computing Systems and Software”.

УДК 004. 4(075.8)

ББК 32.973.26-018 я73

ISBN 978-601-7537-23-4

CONTENTS

Introduction

One day one man wanted to change the whole world for better. After some time, he understood that the world is too big for him to change, so he decided to change his country. After some more time passed, he understood that his country is also too big. That's why he decided to change his city. But later he again revised his aim to change only street.

Finally, at the end of his life he understood that he should have changed himself first, then perhaps he could have changes others, but it was too late.

There are two things that we don't respect enough in our life: our free time and health.

What is time? Time is 1 second or millisecond. These milliseconds produce minutes, hours, days, weeks, years and our life. So when we do any job we must know that it will be better if we do it in small parts but continuously.

When we use the principles of Microlearning approach in our everyday life, we actually do not think about it. This book was prepared for students, who wants to learn C++ programming language by using these Microlearning principles that can be easily observed in daily life.

I would like to express my sincere gratitude to my advisor Prof. Niyazi Ari for the continuous support of my book, for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me in all the time of research and writing of this book. I could not have imagined having a better advisor and mentor for my book.

General concept of Microlearning in C++ programming language

General concept of Microlearning in C++ is illustrated in figure 1. Where general topics means mostly known topics of C++ programming languages such as arrays, pointers e.t.c. Each general topic consists of sub topics and all of them are under list of topics.

Each item from this topic has its own web page, which is constructed in 3 parts: description, C++ code and problems.

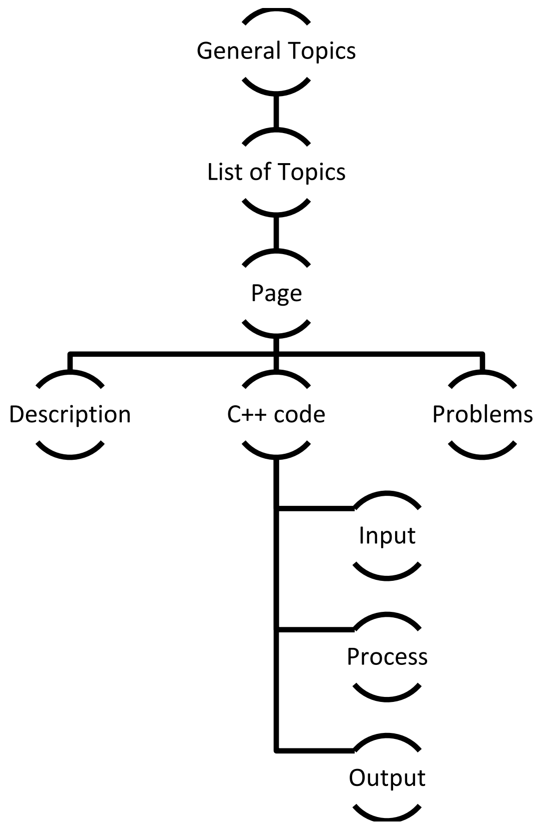

Figure 1 General Concept of Microlearning of PL C++

Description part explains own topic definitely, shortly and unique. It follows C++ code with comments, which includes three parts input, process and output as shown in figure 2.

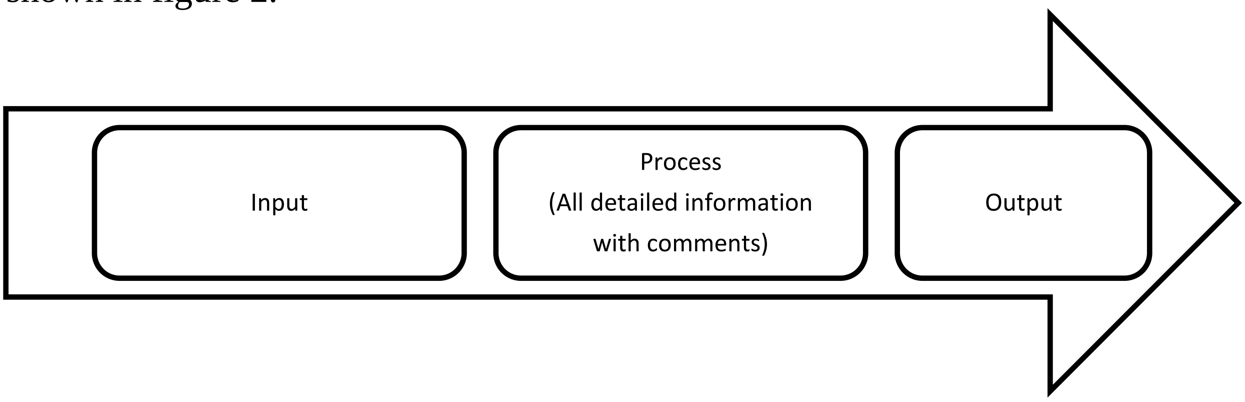

Figure 2 C++ code structure

Topic: Simple program in C++

This is a simple C++ program. Each C++ program contains function int main() { }. If it is needed user enter library by “#include” keyword then the name of library. Also in C++ there is a Standard Library. To use its functions we put “std::” keyword for each function. In future it will show how to make more easy way.

C++ CODE	RESULT

#include <iostream>
 	
int main()
{
std::cout << "First program";
return 0;
}
	First Program

Problems

1. Write a C++ program that prints on screen “Second program is coming”
2. Write a C++ program that prints on screen “Welcome to Kazakhstan”

Topic: Compile and execute

In C++ there are two ways of writing programs with graphical user interface (mostly it is called IDE) or with black screen (mostly it is called command prompt, console, terminal). For second way algorithm is

1 ¬ Create a directory (folder), then create a file with cpp extension and open in editor (notepad in Windows).

2 – Write C++ code. (Also can be copied)

3 – Open command prompt (in Windows run Windows sign + R then write cmd)

4 – Go to needed folder by using command cd.

5 – In command prompt write “g++ name_of_file.cpp” and press Enter

6 – If no error appears name_of_file.exe or a.exe must appear.

7 – Write this file and press Enter

C++ CODE	RESULT

#include <iostream>
 	
int main()
{
std::cout << "Second program";
return 0;
}
	Second Program

Problem

1. Write a C++ program with output “Hello”

Topic: Comments

In C++ there are 2 types of comments

1 – single line comment ¬ //

2 – multiline comment ¬ /* ...

…......

…. */

Mostly new programmers do not use comments in their programs. But after more practice and time they understand that it is important. Because humans have ability to forget.

C++ CODE	RESULT

// Comments in C++
#include <iostream>
 	
int main()
{
int i; // single-line comment
/* Read the number
to be tested whether odd or even
*/
std::cout << "Enter an integer : ";
std::cin >> i;
// check if number i is even or odd
if ((i%2)==0)
std::cout << i <<" is even";
else
std::cout << i <<" is odd";
return 0;
}
	Try 1:
	Enter an integer : 5
	5 is odd
	
	Try 2:
	Enter an integer : 6
	6 is even

Problems

1. Write a C++ program with comments that checks if number is divisible by 3.

2. Write a C++ program of adding two float numbers with comments.

3. Write a C++ program with comments that checks if number is positive or negative.

Topic: input/output

In C++ there are two mostly used input/output operators cin/cout. iostream library must be included to use input/output operators.

In C++ there is a Standard Library. In this library there are a lot of ready functions. To use them std keyword (in most case it is called namespace) is used.

C++ CODE	RESULT
#include <iostream> // library for input/output	Try 1:
int main()	3
{	3
int i; // new integer is defined	
std::cin >> i; // reading integer from keyboard	Try 2:
std::cout << i; // showing integer to display	25
return 0;	25
}	

Problems

1. Write a C++ program that receives two integer numbers and shows their product

2. Write a C++ program that receives two double numbers and shows their division

3. Write a C++ program that receives one integer and one double numbers and shows their sum

Topic: Data types

In C++ there are fundamental data types for storing variables in computer's memory. The memory in computers is organized in bytes.

Int, long char are examples of data types

C++ CODE

```cpp
#include <iostream>

int main()
{
int a;
char c;
std::cout << "Enter number :";
std::cin >> a;
std::cout << "Enter a character :";
std::cin >>c;
std::cout << a << " " << c;
return 0;
}
```

RESULT

```
Try 1:
Enter number :5
Enter a character :d
5 d
Try 2:
Enter number :12
Enter a character :Z
12 Z
```

Problems

1. Write a C++ program that receives two integer numbers and shows their sum

2. Write a C++ program that receives two double numbers and shows their sum

3. Write a C++ program that receives three characters and print them in in one line

Topic: Data type of integer

Integers are variables used to represent whole numbers. In C++ there are int, short, long. Also it can be signed and unsigned.

Unsigned int means positivie integers. By default integer types are signed. Also in C++ int can be declared in one line as int a,b,c.

C++ CODE

```
#include <iostream>

int main()
{
int a1, a2;
long l1, l2;
unsigned short s1;
std::cout << "Enter numbers :";
std::cin >> a1 >> a2;
std::cout << "Enter long numbers :";
std::cin >>l1 >> l2;
std::cout << a1 + a2 << " " << l1 + l2;
return 0;
}
```

RESULT

```
Try 1:
Enter numbers : 20 40
Enter long numbers : 1000 1
60 1001
Try 2:
Enter numbers : 1 2
Enter long numbers : 100000
1
3 100001
```

Problems

1. Write a C++ program of division of 2 short integers

2. Write a C++ program of product of 3 signed 3 long integers

3. Write a C++ program of product of long and integer

Topic: Data type of real numbers

Variables of type “float”, “double” are used to store real numbers. For them “+3.5”, “3.5” and “3.500” all represent the same value.

In C++ “.” sign is used, not “,” for real numbers.

Real numbers can be written in floating¬point form (scientific notation). For example 0.35 could be written as “3.5e+01” (i.e. 0.35 x 10).

C++ CODE	RESULT
`#include <iostream>`	Try 1:
	Enter numbers :3.5 6.7
`int main()`	Enter numbers : 4.777 -9.76
`{`	10.2 -4.983
`double a1, a2;`	Try 2:
`float l1, l2;`	Enter numbers :1.1 2.22
`std::cout << "Enter numbers :";`	Enter numbers : 4 -9
`std::cin >> a1 >> a2;`	3.32 -5
`std::cout << "Enter numbers :";`	
`std::cin >>l1 >> l2;`	
`std::cout << a1 + a2 << " " << l1 + l2;`	
`return 0;`	
`}`	

Problems

1. Write a C++ program that shows multiplication of 2 double numbers

2. Write a C++ program that shows division of 2 float numbers

3. Write a C++ program that shows summation of 3 float numbers written in scientific notation.

Topic: bool data type

Bool data type can take only 2 values true or false. Also in C++ bool data types false is equal to 0 and true is equal to 1.

C++ CODE

```cpp
#include <iostream>

int main()
{
bool b1=false, b2;
int a1;
std::cout << "Enter number :";
std::cin >> a1;
if (a1>0) b1=true;
if (b1) std::cout << "More than 0 \n";
// same as if (b1==true)
b2=0; // same as b2 = false
std::cout << b1 << " " << b2;
return 0;
}
```

RESULT

```
Try 1:
Enter number :4
More than 0
1 0
Try 2:
Enter number :-4
0 0
```

Problems

1. Write a C++ program that print “101” with using bool.
2. Write a C++ program that checks if number even or odd with using bool

Topic: character

In C++ char data type is used for a single character. There are standard characters like upper and lower case letters of English alphabet, the digits '0'..'9' and some special symbols such as *, @, !, etc. Collection of characters is ASCII.

Char are enclosed in single quotation mark ('). For example, in C++ char c='a'; works fine, but char c=“a”; is compilation error.

Characters are represented as integers. (char is a subset of int). For example, in ASCII '9' is decimal 57 and 'A' is decimal 65.

C++ CODE

```
#include <iostream>

int main()
{
char b1, b2;
std::cout << "Enter characters :";
std::cin >> b1 >> b2;
std::cout << int(b1) << " " << int(b2) << "\n";
std::cout << b1 << "-" << b2 << "=" <<b1 - b2;
return 0;
}
```

RESULT

```
Try 1:
Enter characters : 5 7
53 55
5-7=-2
Try 2:
Enter characters : Z A
90 65
Z-A=25
```

Problems

1. Write a C++ program that shows a representing character of given integer

2. Write a C++ program that receives a character and print if it is digit, letter or special symbol

Topic: Type casting

When one int number is divided by another int number answer is int number. To make result double type casting is used.

There are 2 ways:

answer = (double) (numerator) / denominator

answer = static_cast<double>(numerator) / denominator

C++ CODE	RESULT
#include <iostream>	Try 1:
	Enter numbers : 5 2
int main()	5/2=2
{	5/2=2.5
int b1, b2;	5/2=2.5
double d1, d2, d3;	Try 2:
std::cout << "Enter numbers :";	Enter numbers : 12 4
std::cin >> b1 >> b2;	12/4=3
d1 = b1 / b2;	12/4=3
d2 = (double) (b1) / b2;	12/4=3
d3 = static_cast<double> (b1) / b2;	
std::cout << b1 << "/" << b2 << "=" << d1 << "\n";	
std::cout << b1 << "/" << b2 << "=" << d2 << "\n";	
std::cout << b1 << "/" << b2 << "=" << d3 << "\n";	
return 0;	
}	

Problem

1. Write a C++ program that receives 2 double numbers, divide them by 3 and show result in int type.

Topic: Scope of variables

A variable can be either of global or local scope.

A global variable is a variable declared in the main body of the source code, outside all functions, while a local variable is one declared within the body of a function or a block. If there is same name for global and local variable “::” keyword is used for global.

C++ CODE	RESULT

<pre>#include <iostream>
 	
int x1=5, x2=10; // global variables
 	
int main()
{
int x1;   // local variables
std::cout << "Enter number :";
std::cin >> x1;
std::cout << "local x1 : " << x1 << "\n";
std::cout << "global x1 : " << ::x1 << "\n";
std::cout << "global x2 : " << x2 << "\n";
return 0;
}</pre>
	Try 1:
	Enter number : 8
	local x1 : 8
	global x1 : 5
	global x2 : 10
	Try 2:
	Enter number : 14
	local x1 : 14
	global x1 : 5
	global x2 : 10

Problem

1. Write a C++ program that receives 3 integers. For each of them add global int x, which equals to 5.

Topic: Constants

Constant variables are variable with fixed value and cannot be changed.

Its structure is

const data_type name_of_variable = value;

Note: constant variables must be initialized.

	C++ CODE	RESULT
1	#include <iostream>	compilation error
2		Try 1:
3	int main()	line 5 : error: unitialized const pi
4	{	
5	const double pi; // error, it must be initialized	
6	const int x2 = 5; // no error	
7	int x3;	
8	std::cout << "Enter radius :";	
9	std::cin >> x3;	
10	std::cout << "Area : " << pi * x3 * x3 << "\n";	
11	return 0;	
12	}	

Problems

1. Modify a C++ program above to compile correctly.

2. Write a C++ program that calculates volume of pyramid with using constant variable.

Topic: Define

In C++ there is a keyword “#define” to define specific name for constants that is used often. Also it is called macros. Its structure is:

#define identifier value

Note: do not use define in big projects, because it may cause to misunderstandable errors in team.

C++ CODE	RESULT
#include <iostream>	Try 1:
	Enter radius : 2
#define pi 3.14	6.18
	Try 2:
int main()	Enter radius : 1
{	3.14
int x3; // radius	
std::cout << "Enter radius :";	
std::cin >> x3;	
std::cout<<x3*pi << " \n";	
return 0;	
}	

Problem

1. Write a C++ program that calculating area of circle with defined pi.

Topic: Enum

Enumeration is used to give a name as value to int data type. For example in week 7 days. It can be written as

enum Week { MON=1, TUE, WED, THU, FRI, SAT, SUN=0}. By this way it is more easy to make understanble code.

By default, members of “enum” are given the values 0, 1, 2, etc., but it can be initialised. Also it is compilation error to write Week = 1;

C++ CODE	RESULT
// in shop at saturday there is a discount 20%	Try 1:
// at sunday discount 40%	TODAY 0
// no discount at other days	20
#include <iostream>	Pay : 12
	Try 2:
int main()	TODAY 0
{	24
enum Week {MON=1, TUE, WED, THU, FRI, SAT, SUN=0};	Pay : 14.4
Week w;	
w = SUN; // lets assume that today is sunday	
std::cout << "TODAY " << w << "\n";	
int price;	
std::cin >> price;	
if (w == SAT)	
std::cout << "Pay : " << price - price*0.2;	
else if (w == SUN)	
std::cout << "Pay : " << price - price*0.4;	
else std::cout << "Pay : " << price;	
return 0;	
}	

Problems

1. Modify a C++ program above with another days with different discounts.

2. Write a C++ program for Traffic Light. There is an enumeration with 3 colors (red, yellow, green).

Topic: Arithmetic operators

There are 5 standard arithmetic operators in C++.

Operation	Arithmetic operator
Addition	+
Subtraction	-
Multiplication	*
Division	/
Modulus	%

C++ CODE	RESULT

// Calculator
 	
#include <iostream>
 	
int main()
{
int x, y;
std::cout << "Enter two numbers : ";
std::cin >> x >> y;
std::cout << "Sum : " << x + y;
return 0;
}
	Try 1:
	Enter two numbers :
	1 5
	Sum : 6
	Try 2:
	Enter two numbers :
	2 3
	Sum : 5

Problem

1. Make a calculator program for 5 operations

Topic: Short writing operators

Mostly for short writing code compound assignment operators are used. Compound assignment operators are operators that contain arithmetic operator and assignment operator (=). For example “x = x + 5;” can be written as “x += 5;”

Also for short writing code increase (++) operators or decrease (++) operators are used. For example “x = x + 1;” can be written as “x++;”

C++ CODE	RESULT
// Calculator	Try 1:
#include <iostream>	Enter two numbers : 4 5
	Sum : 9
int main()	Try 2:
{	Enter two numbers : 3 6
int x, y;	Sum : 9
std::cout << "Enter two numbers : ";	
std::cin >> x >> y;	
x += y; // same as x = x + y	
std::cout << "Sum : " << x;	
return 0;	
}	

Problem

1. Write a C++ program calculator program.

Topic: Equality and relational operators

For decision there are equality and relational operators in C++.

Relational operator	Meaning
>	x is greater than y
<	x is less than y
>=	x is greater than or equal to y
<=	x is less than or equal to y
Equality operator	
==	x is equal to y
!=	x is not equal to y

Note: Mostly they are used with keyword “if”.

C++ CODE	RESULT
// Comparator	Try 1:
#include <iostream>	Enter two numbers : 1 2
	1 is less than 2
int main()	Try 2:
{	Enter two numbers : 2 1
int x, y;	2 is less than 1
std::cout << "Enter two numbers : ";	
std::cin >> x >> y;	
if (x > y)	
std::cout << x << " is more than " << y;	
if (x < y)	
std::cout << x << " is less than " << y;	
return 0;	
}	

Problems

1. Modify a C++ program above for all relational operators.

Topic: logical operators

There are 3 logical operators in C++: “and” (&&), “or” (||) and “not” (!).

a	b	a && b	a	b	a || b
true	true	true	true	true	true
true	false	false	true	false	true
false	true	false	false	true	true
false	false	false	false	false	false

Not operator change true to false and otherwise.

C++ CODE

```cpp
// there are 4 quarters in coordinate system
// by using AND find in which quarter is given coordinates
#include <iostream>

int main(){
int x, y;
std::cout << "Enter two coordinates : ";
std::cin >> x >> y;

// in C++ && are same as and
if ((x>0) && (y>0)) std::cout << "in 1 quarter";
if ((x<0) && (y>0)) std::cout << "in 2 quarter";
if ((x<0) and (y<0)) std::cout << "in 3 quarter";
if ((x>0) and (y<0)) std::cout << "in 4 quarter";

return 0;
}
```

RESULT

```
Try 1:
Enter two coordinates : 1 1
in 1 quarter
Try 2:
Enter two coordinates : -1 -1
in 3 quarter
```

Problems

1. Write a C++ program that receives 3 integers as size of triangle and finds which type of triangle is it.

2. Write a C++ program that checks if it is weekend or workday for given n.

Topic: Operator precedence

There is an operation precedence (which operation will be calculated first) in C++ as in algebra. It is generally same as followed in algebra. Simple algorithm is:

Firstly the operations in parathensis are evaluated. If there are nested parathensis the inner is evaluated first.

Secondly the multiplication, division and modulus are evaluated. If there are two or more operations they are evaluated from left to right.

At last addition and subtraction are evaluated. Here also the rule from left to right is working.

In C++ also can be written crazy expression like “a>>b+a=d%3+(5==3)+c” and it will work if there are variables a,b,c,d.

C++ CODE

```
// Operator precedence
// in comments order numbers of operations
#include <iostream>

int main()
{
int x, y;
std::cout << "Enter two numbers : ";
std::cin >> x >> y;
int z;
z = ((x-y)+(x*x-y)) * (x+y);
// 7    1   4   2  3    6    5
std::cout << z << "\n";

z = ((x==y) + (x%4))+y;
// 5      1      3     2     4
std::cout << z;
return 0;
}
```

RESULT

```
Try 1:
Enter two numbers : 5 7
192
8
Try 2:
Enter two numbers : 6 6
360
9
```

Problems

1. Write a C++ program that finds distance between two points. Hint: use function sqrt from library <cmath>

2. What will be ouput of expression like “z = ((x=y)+(x++)) + (x!=y);” write the operation precedence order.

Topic: Keywords

In C++ there are some definite keywords. The name of variables cannot be as keyword. Also C++ is case sensitive, so there is a difference between return and RETURN.

List of keywords are:

asm, auto, bool, break, case, catch, char, class, const, const_cast, continue, default, delete, do, double, dynamic_cast, else, enum, explicit, export, extern, false, float, for, friend, goto, if, inline, int, long, mutable, namespace, new, operator, private, protected, public, register, reinterpret_cast, return, short, signed, sizeof, static, static_cast, struct, switch, template, this, throw, true, try, typedef, typeid, typename, union, unsigned, using, virtual, void, volatile, wchar_t, while

Additionally :

and, and_eq, bitand, bitor, compl, not, not_eq, or, or_eq, xor, xor_eq

C++ CODE	RESULT
#include <iostream>	Try 1:
	Compilation error
int main()	
{	
int x, y;	
std::cout << "Enter two numbers : ";	
std::cin >> x >> y;	
int for;	
for = x + y; std::cout << for;	
return 0;	
}	

Problems

1. Modify a C++ program above to compile correctly.

Topic: Escape codes

In C++ there are escape codes, which are used to express more additional symbols or actions. Its structure consists of “\” sign and letter.

\n newline

\r carriage return

\t tab

\v vertical tab

\b backspace

\f form feed (page feed)

\a alert (beep)

\' single quote (')

\" double quote (")

\? question mark (?)

\\ backslash (\)

C++ CODE	RESULT
#include <iostream>	Try 1:
	Enter two chars : a W
int main()	a ? ' W Try 2:
{	Enter two chars : 2 z
char ch1, ch2;	2 ? ' z
std::cout << "Enter two chars : ";	
std::cin >> ch1 >> ch2;	
std::cout << "----------- \n";	
std::cout << ch1 << "\t \? \' \t" << ch2;	
return 0;	
}	

Problem

1. Write a C++ program that prints the text in double quotes.

Topic: Keyword if

In most programming language there is a decision keyword if. It is used to testing a condition. Its structure is :

if (condition or conditions) C++ code OR

if (condition or conditions)

{ C++ codes }

There can be used logical operations for more complex condition(s).

C++ CODE	RESULT

#include <iostream>
 	
int main()
{
char ch1;
std::cout << "Enter char : ";
std::cin >> ch1;
if (ch1=='a')
   std::cout<< "You entered letter a";
if (ch1!='a')
   std::cout<< "You do not entered letter a";
return 0;
}
	Try 1:
	Enter char : a
	You entered letter a
	Try 2:
	Enter char : d
	You do not entered letter a

Problems

1. Write a C++ program that tests if given number even or odd.

2. Write a C++ program that tests if given number is 2 digits or 3 digits. (Hint: use module operator “%”)

Topic: if … else structure

In C++ for decision can be used if .. else structure. Its structure is :

if (condition) code

else if (condition) code else code

Note if there are 2 or more statements in code part use “{}” brackets.

C++ CODE

```
#include <iostream>

int main()
{
char ch1;
std::cout << "Enter char : ";
std::cin >> ch1;
if (ch1>='a' and ch1<='z')
{
  std::cout<< "You entered letter \n";
  std::cout<< "And it is lower case";
}
else if (ch1>='A' && ch1<='Z')
  std::cout<< "You entered capital letter";
else if (ch1>='0' && ch1<='9')
  std::cout<< "You entered digit";
else
{
  std::cout<< "You entered symbol";
  char ch2 = ch1;
}
return 0;
}
```

RESULT

```
Try 1:
Enter char : a
You entered letter
And it is lower case
Try 2:
Enter char : `
You entered symbol
```

Problems

1. Write a C++ program that checks if given point inside of triangle or outside.
2. Write a C++ program that for given two squares which of them big or equal

Topic: Difference between = and ==

Mostly new programmers in C++ or programmers from another language have confuse with = and == in operators.

Mostly this type of logical error happens in if structure. Lets assume that in C++ code it is written "if

(n=5) " and "if (n==5)".

In first case number n becomes to equal to 1. (Now number n is equal to 5) In second case number, n checks with 5 and return 0 (true) or 1 (false).

C++ CODE	RESULT
`#include <iostream>`	Try 1:
	Enter two numbers : 2 3
`int main()`	2 is not equal to 5
`{`	5 is equal to 5
`int n1, n2;`	
`std::cout << "Enter two numbers : ";`	Try 2:
`std::cin >> n1 >> n2;`	Enter two numbers : 9 9
	9 is not equal to 5
`if (n1==5) // checks n1 and 5`	5 is equal to 5
`  std::cout<< n1 << " is equal to 5 \n";`	
`else`	
`  std::cout<< n1 << " is not equal to 5 \n";`	
`if (n2=5) // assigns n2 to 5`	
`  std::cout<< n2 << " is equal to 5 \n";`	
`else`	
`  std::cout<< n2 << " is not equal to 5 \n";`	
`return 0;`	
`}`	

Problems

1. Write a C++ program that tests if it leap year or not for given n

2. Write a C++ program that tests if character is capital letter 'A' or not for given character.

Topic: Keyword conditional operator ?

In C++ it can be used conditional operator “?” instead of if .. else structure. Its structure is :

condition ? result1 : result2;

Also conditional operator can be used in cout stream, which is very powerfull feature.

C++ CODE	RESULT
`#include <iostream>`	Try 1:
	Enter two numbers : 1 4
`int main()`	4 is maximum
`{`	1 is minimum
`int n1, n2;`	Try 2:
`std::cout << "Enter two numbers : ";`	Enter two numbers : 8 2
`std::cin >> n1 >> n2;`	8 is maximum
`int max;`	2 is minimum
`max = (n1 > n2) ? n1 : n2;`	
`std::cout<< max <<" is maximum \n";`	
`std::cout<< ((n1<n2) ? n1:n2) << " is minimum";`	
`return 0;`	
`}`	

Problems

1. Write a C++ program that finds maximum for given 3 numbers using conditional operator

Topic: Using file in programs

Mostly in practice it is easy to use files to read and write. Cause sometime it is a waste of time to write again and again some variables to check a program. Also programmer can combine such that he can read from file and show result in display.

Its structure is:	RULES

#include <fstream>
using namespace std;
int main()
{
ifstream fin("name_of_file");
ofstream fout("name_of_file");
fin>>some_data; fout<<some_data;
fin.close(); fout.close(); return 0;
}
	1. Instead of <iostream> <fstream> is included.
	2. In folder where program code is input file must be created.
	3. Instead of cin, cout use fin, fout
	4. At end close them
	Note: do not write in program something like “enter two numbers” because program reads from file.

Note: Instead of using std::fin or std::cin it can be written using namespace std. It is more effective to use.

C++ CODE	RESULT
#include <fstream>	Try 1:
using namespace std;	in input.txt file
int main()	3 4
{	in output.txt file
int n1, n2;	4 is maximum
ifstream fin("input.txt");	3 is minimum
ofstream fout("output.txt");	
fin >> n1 >> n2;	
int max;	
max = (n1 > n2) ? n1 : n2;	
fout<< max <<" is maximum \n";	
fout<< ((n1<n2) ? n1:n2) << " is minimum";	
fin.close();	
fout.close();	
return 0;	
}	

Problems

1. Write a C++ program that reads from file “digit.txt” 15 numbers and saves their sum in file “output.txt”

Topic: for loop

The “for” loop's structure is:

for (initialization; condition; increase)

{ C++ code }

It works in 4 steps. First and second steps are finished by “;”. These steps are:

1 – initialization is executed. This is executed only once.

2 – condition is checked. If it is true the loop continues, else the loop stops and code is not executed.

3 – code is executed. If it is more than 1 statements it must be in “{}”.

4 – increase is executed and the loop go to step 2.

C++ CODE	RESULT
#include <iostream>	Try 1:
using namespace std;	5
	1 2
int main()	2 4
{	3 6
int n1, n2;	4 8
cin >> n1;	5 10
for (int i=1; i<=n1; i++)	Try 2:
{	3
cout << i << " " << i*2;	1 2
cout << endl; // newline same as cout<<"\n";	2 4
}	3 6
return 0;	
}	

Problems

1. Write a C++ program that calculates the sum of first n numbers

2. Write a C++ program that shows the numbers from n to 1.

3. Write a C++ program that prints all characters. Hint: characters are integers in ASCII code.

Topic: while loop

The “while” loop's structure is: while (condition(s) is(are) true) { code }

In while loop there must be increment (decrement) operations or some checking to make condition false, otherwise it can infinite loop.

Difference between for and while is in for loop the number of steps are definite, in while loop the number of steps are undefinite and up to condition.

C++ CODE	RESULT

#include <iostream>
using namespace std;
 	
int main()
{
long int n1, n2, n3=0;
cin >> n1;
n2 = n1;
while (n1!=0)
{
n3 = n3 * 10 + (n1%10);
n1 = n1 / 10;
}
if (n3 == n2)
cout << n3 << " is palindrome of " << n2;
else
cout << n3 << " is not palindrome of " << n2;
return 0;
}
	Try 1:
	135
	531 is not palindrome of 135
	Try 2:
	135531
	135531 is palindrome of 135531
	Try 3:
	1234567654321
	-1126087180 is not palindrome of 2147483647
	(NOTE here error appears because given number is more than long int range)

Problems

1. Write a C++ program that calculates the number of digits in given number n.

2. Write a C++ program that tests if given number is prime or not.

3. Write a C++ program that print all divisors for given number.

Topic: do … while loop

The “do .. while” loop's structure is do { code } while (condition[s] is[are] true);

It works like “while” loop. Also can stay in inifinite loop if programmer forgets to change conditions to false in code.

The main difference between do/while and while loops is that sometimes in do/while it can be 1 more step, because firstly it run code then checks the condition.

C++ CODE	RESULT

#include <iostream>
using namespace std;
 	
int main()
{
long int n1, n2, n3=0;
cin >> n1; n2 = 10; do
{
  cout << "A";
  n2++;
} while (n2<n1);
 	
cout << endl;
n2 = 10;
while (n2<n1)
{
  cout << "A";
  n2++;
}
return 0;
}
	Try 1:
	15
	AAAAA
	AAAAA
	
	Try 2:
	5
	A

Problems

1. Write a C++ program that prints all prime number till given number.

2. Write a C++ program that prints as shown below till given number n=3

1

22

333

Topic: break

Break command is used to finish loop and exit from loop even if the loop condition is true.

C++ CODE	RESULT

#include <iostream>
using namespace std;
 	
int main()
{
int n1;
cin >> n1;
for (int i=n1; i>0; i--)
{
cout << i << " ";
if (i==4)
{
cout << "Break";
break;
cout << "STOP";
}
}
return 0;
}
	Try 1:
	8
	8 7 6 5 4 Break
	Try 2:
	11
	11 10 9 8 7 6 5 4 Break

Problems

1. Write a C++ program that finds a biggest prime number for given number using break.

2. Write a C++ program that receives two numbers (x, y) and prints all numbers between them. If there is a number between x and y that is equal to sum of x and y the loop breaks.

Topic: continue

The continue causes to pass rest of loop and go to next iteration.

	C++ CODE	RESULT
1	#include <iostream>	Try 1:
2	using namespace std;	9
3		9 8 7 6 5 Continue 3 2 1
4	int main()	Try 2:
5	{	11
6	int n1;	11 10 9 8 7 6 5 Continue 3 2 1
7	cin >> n1;	
8	for (int i=n1; i>0; i--)	
9	{	
10	if (i==4)	
11	{	
12	cout << "Continue ";	
13	continue;	
14	cout << "STOP ";	
15	}	
16	cout << i << " ";	
17	}	
18	return 0;	
19	}	

Problems

1. Write a C++ program that prints all odd numbers till given number

Topic: switch

The “switch” is needed to use if .. else in more easy way. Its structure is :

switch (variable) {

case 1 :

// Process for variable = 1

... break;

case 5 :

// Process for variable = 5

... break;

default :

// Process for all other cases.

...

}

Note: after each case break must be written, in other case it passes to other cases (see try 3 in example)

C++ CODE	RESULT
#include <iostream>	Try 1:
using namespace std;	Enter your grade : 3
int main()	Bad
{	Try 2:
int n1;	Enter your grade : 4
cout << "Enter your grade : ";	Good
cin >> n1;	Try 3:
switch (n1)	Enter your grade : 5
{	Very goodGood
case 5 : cout << "Very good";	
//break;	
case 4 : cout << "Good"; break;	
case 3 : cout << "Bad"; break;	
case 2 : cout << "Very bad"; break;	
default: cout << "You must enter number between 2 and 5"; break;	
}	
return 0;	
}	

Problems

1. Write a C++ program that shows month name up to given integer

Topic: exit

The “exit” is used to terminate program even if it is not finished. It can be used when error happens in code.

C++ CODE	RESULT

#include <iostream>
#include <cstdlib>
using namespace std;
 	
int main()
{
int n1, n2;
cout << "Enter two numbers : ";
cin >> n1 >> n2;
if (n2 == 0) exit(1);
int div = n1 / n2;
cout << div;
return 0;
}
	Try 1:
	Enter two numbers :
	6 3
	2
	Try 2:
	Enter two numbers :
	6 0

Problems

1. Write a C++ program that calculates a square root by using delta. Terminate program if delta is less than 0.

Topic: Pointer

A pointer is a variable that holds the address of another variable. Pointer also can be understood as remote¬control for TV. Its structure is

data_type * name_of_pointer;

An address of operator (&) sign is used to take an address of variable.

C++ CODE	RESULT

#include <iostream>
using namespace std;
int main()
{
int intStandart;
int *intPointer;
cin >> intStandart;
intPointer = &intStandart;
cout << "Value " << intStandart << endl;
cout << "Address of standart "
        << &intStandart << endl;
cout << "Value of pointer " << intPointer << endl;
cout << "Address of pointer " << &intPointer << endl;
return 0;
}
	Try 1:
	12
	Address of standart 0xbffc666c
	Value of pointer 0xbffc666c
	Address of pointer 0xbffc6668
	
	Try 2:
	22
	Address of standart 0xbffc666c
	Value of pointer 0xbffc666c
	Address of pointer 0xbffc6668
	
	Note: address can be changed up to system

Problems

1. Write a C++ program that modify program above with usage of double.

2. Write a C++ program that contains receives integer x and makes 2 pointers ptr1 and ptr2 to integer x. Print values of x, ptr1 and ptr2.

Topic: Dereference operator (*)

Pointer means address of another variable. To show value of pointed variable the dereference operator (*) is used.

Do not mix * sign with dereference operator and pointer. Dereference operator is used after pointer is declared.

C++ CODE	RESULT
#include <iostream>	Try 1:
using namespace std;	24
int main()	Value 24
{	Value 24
int a;	Value after change 48
// here * is used for pointer	
int *ptr1=&a, *ptr2=&a;	Try 2:
cin >> a;	4
cout << "Value " << a << endl;	Value 4
cout << "Value " << *ptr1 << endl;	Value 4
	Value after change 8
// here * is used as dereference	
*ptr2 = 2 * a;	
cout << "Value after change" << *ptr1 << endl;	
return 0;	
}	

Problems

1. Write a C++ program that swaps two integers with using pointer.

2. Write a C++ program that shows sum of 2 number with using pointers.

3. Write a C++ program that receives integer x. Then makes a pointer to x and change its value. In display show value of x, value of pointer, value of pointer by using dereference.

TOPIC: Size of pointers

A size of pointer is 4 bytes or 8 bytes, because of architecture of computer. It does not change up to data type of pointer. So lets reminder again “pointer saves address of variable not its value”.

C++ CODE	RESULT
#include <iostream>	Try 1:
using namespace std;	a
	1.3
int main()	1
{	4
char a;	8
double b;	4
char *ptr1=&a; // pointer for integer double	
*ptr2=&b; // pointer for double	
cin >> a;	Try 2:
cin >> b;	Q
cout << sizeof(a) << endl; // size of char	2
cout << sizeof(ptr1) << endl; // size of pointer	1
cout << sizeof(b) << endl; // size of double	4
cout << sizeof(ptr2) << endl; // size of pointer	8
	4
return 0;	
}	

Problems

1. Write a C++ program that shows sizes of int, long int, float data types and sizes of pointers to them.

TOPIC: Pointer to pointer

In C++ there is pointer to pointer which means that if there is a pointer, programmer can make another pointer to it. In this case add an asterisk (*) for each level of reference in declarations. Its structure is

data_type *pointer1;

data_type **pointer2 = &pointer1; and so on...

C++ CODE	RESULT
#include <iostream>	Try 1:
using namespace std;	24
	Value1 : 24
int main()	Adress1 : 0xbfb4e9ec
{	Value2 : 0xbfb4e9ec
int a; //integer a	Adress2 : 0xbfb4e9e8
int *ptr1=&a; // pointer to a	Value3 : 0xbfb4e9e8
int **ptr2=&ptr1; // pointer to ptr1	Adress3 : 0xbfb4e9e4
cin >> a;	Note : the address may
	not be same in testing
cout << "Value1 : " << a << " Address1 : " << &a << endl;	
cout << "Value2 : " << ptr1 << " Address2 : " << &ptr1 << endl;	
cout << "Value3 : " << ptr2 << " Address3 : " << &ptr2 << endl;	
return 0;	
}	

Problems

1. Write a C++ program that receives char, double and long int data types and then make triple pointer to pointer.

Topic: Array

An array is a data type that saves multiple variables with same type through a single name by use of an index. An index is an integer parameter with use of the subscript operator ([]). In C++ index begins with 0 and finishes with size of array – 1 elements. Index is used to assign to ith¬1 element of array.

Its structure is:

data_type name_of_array [size_of_array];

C++ CODE	RESULT
#include <iostream>	Try 1:
using namespace std;	1
	2
int main()	3
{	4
int a1,a2,a3,a4,a5; //5 integers	5
int a[5]; // array for 5 integers	5 4 3 2 1
for (int i=0; i<5; i++)	Try 2:
cin >> a[i];	5
	4
for (int i=4; i>=0; i--)	3
cout << a[i] << " ";	2
	1
return 0;	1 2 3 4 5
}	

Problems

1. Write a C++ program that makes an array of 20 integers, reads 20 integers then shows odd integers, even integers, positive integers, negative integers and then shows only integer in odd index .

2. Write a C++ program that makes an array of 20 doubles, reads 20 doubles then finds maximum, minimum and prints them with their indexes.

Topic: Loops in array

Mostly arrays are used with loops (for, while). With them there can be operations like:

1) Search for a value (find xin array).

2) Calculate a value (average, total of array).

3) Reorganize the array (sort from lowest to highest)

In array there can be errors like out of range this means that programmer tries to call an index that is more that can be, for example there is an array of 5 elements, programmer writes array [5], and this is error because 5th element of array is array[4].

To prevent this type of error only one way is practice.

C++ CODE	RESULT
`#include <iostream>`	Try 1:
`using namespace std;`	3
	4
`int main()`	2
`{`	1
`int size=5;`	5
`int a[size];       // array for 5 integers`	1 2 3 4 5
`for (int i=0; i<size; i++)`	Try 2:
`  cin >> a[i];`	10
	9
`for (int i=0; i<size-1; i++)`	8
`{`	7
`  for (int j=i+1; j<size; j++)`	6
`  if (a[i]>a[j])`	6 7 8 9 10
`     swap(a[i], a[j]);`	
`     // swap is function for exchange values of two datas`	
`}`	
`for (int i=0; i<size; i++)`	
`  cout << a[i] << " ";`	
`return 0;`	
`}`	

Problems

1. Write a C++ program that sorts 20 numbers from bigger to smaller (descending order)

2. Write a C++ program that has 3 arrays a,b,c of doubles. Arrays a and b are read from file. Then their sum are saved in array c. Print c in monitor.

Topic: Multidimensional array

Multidimensional array is an array that contains array or "arrays of arrays". For example, a matrix array can be imagined as a table of elements with given size as shown below. Its structure is

data_type name_of_array [number_of_row][number_of_column] // int a[2][3]

[0][0] [0][1] [0][2]

[1][0] [1][1] [1][2]

C++ CODE	RESULT
#include <iostream>	Try 1:
using namespace std;	2 2
#define width 5	1
#define height 3	3
int main() {	4
int a,b;	5
cin >> a >> b;	1 3
int matrix[a][b]; // multidimensional array	4 5
for (int i=0; i<a; i++)	Try 2:
for (int j=0; j<b; j++)	3 2
cin >> matrix[i][j];	9
	9
for (int i=0; i<a; i++) {	9
for (int j=0; j<b; j++)	9
cout << matrix[i][j] << " ";	9
cout<<endl; // why we need it here?	9
}	9 9
	9 9
return 0;	9 9
}	9 1

Problems

1. Write a C++ program that multiplies two matrixes (2x2).

2. Write a C++ program that prints multiplication table.

Topic: Relation between pointers and arrays

In C++ a pointer can be equivalent to the address of the first element of array. This can be represented like:

```
int a [20];
int *ptr1;
ptr1 = a; // ptr1 = &a[0]
```

Programmer can use arithmetic rules to go to next elements like ptr1++. Note C+ compiler automatically goes to next element up to data type. This means that if array is double then pointer goes 8 bytes.

C++ CODE	RESULT
#include <iostream>	Try 1:
using namespace std;	a X !
	2
int main()	4
{	8
char ch[3]={'a','X','!'};	0 2
char *ptr2;	1 4
ptr2 = ch;	2 8
cout << *ptr2 << " ";	
cout << *(++ptr2) << " ";	Try 2:
cout << *(++ptr2) << " " << endl;	a X !
	4
int a[3];	16
int *b;	34
int x;	0 4
b = a; // or b = &a[0];	1 16
cin >> x ;	2 34
*b = x;	
cin >> x;	
b++;	
*b = x;	
cin >> x;	
b = a;	
*(b+2) = x;	
for (int i=0; i<3; i++)	
cout << i << " " << a[i] << endl;	
return 0;	
}	

Problems

1. Write a C++ program that counts even integers in array using pointer. Use:

   for (int *ptr = a; ptr < a + 3; ptr++) where a is array and 3 size of array.

Topic: Dynamic memory

Mostly in programs size of memory are definited before execution of the program. If programmer needs to change size of memory during runtime dynamic memory with operators new and delete are used. Dynamic memory allocation allows using memory more dynamically.

int *ptr = new int; // dynamically allocate an integer

*ptr = 7; // assign 7 to this integer

delete ptr; // unallocate memory assigned to another

Note: the delete operator does not delete the pointer — it deletes the memory that the pointer points to!

For arrays new[], delete[] are used.

int *ptr = new int[5]; // dynamically allocate an array where size can be flexible

*ptr[2] = 7; // assign 7 to this 2nd element of array

delete[] ptr; // do not mix with delete

C++ CODE	RESULT
#include <iostream>	Try 1:
using namespace std;	4
	1
int main()	2
{	3
int *a = new int;	4
cin >> *a;	1 2 3 4
int *b = new int[*a];	Try 2:
for (int i = 0; i<*a; i++)	3
cin >> b[i];	-1
	-6
for (int i = 0; i<*a; i++)	8
cout << b[i] << " ";	-1 -6 8
delete a;	
delete[] b;	
return 0;	
}	

Problems

1. Write a C++ program that receives integer n and creates a dynamic array a of n size of doubles, then shows a maximum number in array, then array a is deleted, then again receives integer n and creates a dynamic array a of n size of long int, then shows minimum number in array.

Topic: Vector

Vectors are implemented as dynamic arrays. Vector have their elements stored in contiguous storage and can be accessed as in arrays. Vectors have one important advantage with respect to arrays in C++: vectors can be resized during the execution of the program to accommodate extra elements as needed. To use vector there must be #include <vector> in code

Its structure is:

vector<data_type> name_of_vector;   OR vector<data_type> name_of_vector (size);

By using Standard Library for vector there are ready functions, which makes easy programming. Some of them are:

begin(), end()	The beginning or end of vector
size()	Counts number of elements in vector
resize(int)	Make more bigger or smaller of size of vector
push_back(int), pop_back	Add or delete the element from the end

C++ CODE	RESULT
#include <iostream>	Try 1:
#include <vector>	3
using namespace std;	1
int main() {	3
vector <int> v1;	5
int a;	Size 1 : 3
cin >> a;	Size 2 : 5
v1.resize(a);	1 3 5 80
for (int i=0; i<v1.size(); i++)	Try 2:
cin >> v1[i];	4
cout << "Size 1 : " << v1.size() << endl;	-1
v1.push_back(80);	7
v1.push_back(90);	-4
cout << "Size 2 : " << v1.size() << endl;	5
v1.pop_back();	Size 1 : 4
for (int i=0; i<v1.size(); i++)	Size 2 : 6
cout << v1[i] << " ";	-1 7 -4 5 80
return 0;	
}	

Problems

1. Write a C++ program that makes a vector of student_GPA and receives GPA upon to number of students in class. Then program calculates an average of class.

2. Write a C++ program that shows receives a vector of n double numbers and sorts them.

Topic: Iterators in vector

Iterators provide a flexible way of accessing elements in a vector. They allow referring directly to the elements of the vector and iterating over the entire vector.

Its structure is: vector<int>::iterator i;

As in pointers to access the element "pointed to" by the iterator, * operator is used.

C++ CODE	RESULT
#include <iostream>	Try 1:
#include <vector>	3
using namespace std;	1
	2
int main()	3
{	1 2 3 80
vector <int> v1;	
vector<int>::iterator i;	Try 2:
int a;	2
cin >> a;	-4
v1.resize(a);	5
	-4 5 80
for (i=v1.begin(); i<v1.end(); i++)	
cin >> *i;	
v1.push_back(80);	
for (i=v1.begin(); i<v1.end(); i++)	
cout << *i << " ";	
return 0;	
}	

Problems

1. Write a C++ program that uses iterators to find maximum element in vector.

2. Write a C++ programs that clears all data in vector

Topic: Some algorithms for vector

In most programming languages algorithms are implemented in them as Library. This is called “do not create a bicycle”. In C++ Standart Library there are also algorithms for vectors such as:

Algorithm and its description	Example of use
find - finds a given value. If a value is found an iterator to it returns else it returns end.	if (find (v1.begin(), v1.end(), 25) == v1.end())
sort - sorts the elements in the vector	sort (v1.begin(), v1.end());
reverse - reverses the order of the elements in the vector. sort (v1.begin(), v1.end());	reverse (v1.begin(), v1.end());
replace - replaces all the occurrences of a given with another	replace(v1.begin(), v1.end(), 'a', 'A');

Note1: To use these algorithms in C++ code write #include <algorithm>

Note2: To learn more just google it

C++ CODE

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

int main()
{
vector <int> v1;
int a;
cin >> a;
v1.resize(a);

for (int i= 0; i<v1.size(); i++)
   cin >> v1[i];

sort(v1.begin(),v1.end());
for (int i= 0; i<v1.size(); i++)
   cout << v1[i] << " ";
cout << endl;

reverse(v1.begin(),v1.end());
for (int i= 0; i<v1.size(); i++)
   cout << v1[i] << " ";
return 0;
}
```

RESULT

```
Try 1:
3
1
2
3
1 2 3
3 2 1

Try 2:
4
-4
5
-3
6
-4 -3 5 6
6 5 -3 -4
```

Problems

1. Write a C++ program that searches for definite number in vector.

Topic: Character Sequence

The sequence of characters (arrays of char) produces string. A string is a sequence of characters that are interpreted as a piece of text. To show end of characters a special character is used to signal the end: the null character '\0' (backslash, zero).

```
char  str[]="SDU";

char str [] = { 'S', 'D', 'U', '\0' };
```

0	1	2	3
S	D	U	\0

This type is called C¬style string. Also in C++ there is a special class <string>.

C++ CODE	RESULT

#include <iostream>
using namespace std;
 	
int main ()
{
char init[] = "Enter name: ";
char out[] = "Bye ";
char name[25];
cout << init;
cin >> name;
cout << out << name << "!";
return 0;
}
	Try 1:
	Enter name: Askar
	Bye Askar!
	
	Try 2:
	Enter name: A B
	Bye A!

Problems

1. Write a C++ program that reads a name and surname and shows them splitly.

2. Write a C++ program that reads a name, age and class of student, then prints them.

Topic: Size of Character Sequence

The size of character is equals to text plus 1 (null operator). char str[]="SDU";

0	1	2	3
S	D	U	\0

But if we declare as given size [integer] then size shows the full.

C++ CODE	RESULT
`#include <iostream>`	Try 1:
`using namespace std;`	Enter name: Askar
	13
`int main ()`	E n t e r n a m e :
`{`	25
`char init[] = "Enter name: ";`	A s k a r $ e ^ & # @
`char name[25];`	
`cout << init;`	Try 2:
`cin >> name;`	Enter name :Askar Almaty
	13
`cout << sizeof(init)<<endl;`	E n t e r n a m e :
`for (int i=0; i < sizeof(init); i++)`	25
`   cout << init[i] << " ";`	A s k a r $ e ^ & # @
`cout << endl <<sizeof(name)<<endl;`	Note : symbols can be
`for (int i=0; i < sizeof(name); i++)`	different
`   cout << name[i] << " ";`	
`return 0;`	
`}`	

Problems

1. Write a C++ program that reads a name and show only vowels.

Topic: Buffer

A buffer is temporary memory to hold data. So in C++ by cin programmer temporarily holds the user input. This can be error if user enters more than array can hold. This called a buffer overflow. Buffer overflow results in other memory being overwritten, which usually causes a program crash, but can cause any number of other issues.

The recommended way of reading strings using cin is to use cin.getline(name_of_array, number_of_characters);

This call to cin.getline() will read up to number_of_characters¬1 characters into name_of_array (¬1 is null terminator!). Any excess characters will be discarded. This guarantees that buffer overflow will not occur.

C++ CODE

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

int main ()
{
char init[] = "Enter name :";
char name[25];
cout << init;
  cin.getline(name, 25);

for (int i=0; i < sizeof(name); i++)
{
   if ((name[i]>=97) && (name[i]<=122))
        cout << char(name[i]-'a'+'A') << " ";
   else cout << name[i] << " ";
}

return 0;
}
```

RESULT

```
Try 1:
Enter name :Askar
A S K A R $ e ^ & # @

Try 2:
Enter name :Askar Almaty
A S K A R   A L M A T Y^ & #
@

Note : symbols can be different
```

Problems

1. Write a C++ program that reads a name and surname together with space and shows only name.

2. Write a C++ program that reads some text and converts all capital letters to small letters and all small letters to capitals

3. Write a C++ program that reads 3 cities with their population in one line, then shows city with maximum population

<u>Topic: Functions to manipulate C-style strings</u>

C++ provides many functions to manipulate C-style strings.

strcat(destination, source)	Appends one string to another
strncat(destination, source, n)	Appends one string to another with length n
strcmp(destination, source)	Compare two strings (returns 0 if equal)
strncmp(destination, source, n)	Compare two strings up to a specific number of characters (returns 0 if equal)
strlen(string)	Returns the length of a string (excluding the null terminator)
strcpy(destination, source)	Make a copy of a string to another.
strncpy(destination, source, n)	Make a copy of a string to another with length n

Note to use them #include <cstring> must be in C++ code

C++ CODE

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

int main ()
{
char init[] = "Enter name :";
char name[25];
cout << init;
cin.getline(name, 25);

char last[40];
strcpy(last, name);
cout << last << endl;

strncpy(last, name, 3);
last[3]=0;
cout << last << endl;

for (int i=0; i < strlen(name); i++)
    cout << name[i] << " ";

return 0;
}
```

RESULT

```
Try 1:
Enter name :Askar
Askar
Ask
A s k a r

Try 2:
Enter name :Askar A Askar A
Ask
A s k a r   A
```

<u>Problems</u>

1. Write a C++ program that reads two texts and number n and then compares these texts with n and without n.

2. Write a C++ program that reads two texts and makes a new text that includes these texts.

Topic: String class

In the standard library (std), there is a string class. This std::string lets work with strings in a way that is much easier. Programmer can assign strings to them using the assignment operator and they will automatically resize to be as large or small as needed.

In C++ code there must be #include <string>

C++ CODE

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

int main ()
{
char init[] = "Enter name :";
string name;
cout << init;
cin>>name;

cout << "Hello " << name << endl;
cout << name.length();

return 0;
}
```

RESULT

```
Try 1:
Enter name :Askar
Hello Askar
5

Try 2:
Enter name :A B Hello A
1
```

Problems

1. Write a C++ program that reads a 2 students name and GPA. Then prints student name, whose GPA is higher.

Topic: Reading string

To read full string use getline(cin, name_of_string); Also do not forget that string can work as array.

C++ CODE

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

int main ()
{
char init[] = "Enter name :";
string name;
cout << init;
getline(cin,name);

cout << "Hello " << name << endl;

cout << name.length() << endl;
cout << name[0] << name[1] << name[2] << endl;

return 0;
}
```

RESULT

```
Try 1:
Enter name :Askar
Hello Askar
5
Ask

Try 2:
Enter name :Askar B Hello A
7
Ask
```

Problems

1. Write a C++ program that reads a name and surname in one line. Then program prints them separately.

2. Write a C++ program that reads an arithmetic expression like “5 + 4 + 2” with one digit number and operators + or minus. Calculate result.

Topic: Conversion

Sequences of characters stored in char arrays can easily be converted into string objects just by using the assignment operator:

```
string str1;
char chr1[]="SDU";
str1 = chr1;
```

C++ CODE	RESULT

#include <iostream>
#include <string>
using namespace std;
 	
int main ()
{
char init[] = "Enter name :";
string name, text;
text = init;
 	
cout << text;
getline(cin,name);
 	
cout << text << name << endl;
 	
return 0;
}
	Try 1:
	Enter name :Askar
	Enter name :Askar
	
	Try 2:
	Enter name :A B Enter name :A B

Topic: Some string functions

There are some functions which help to work with strings.

Name	Description
length	Length of string
clear	Clear string
append	Append to string
push_back	Append a character to string
insert	Insert into string
replace	Replace part of string
find	Find content in string

C++ CODE	RESULT

#include <iostream>
#include <string>
 	
using namespace std;
 	
int main ()
{
string str1, str2;
 	
cin >> str1 >> str2;
swap(str1, str2);
cout << str1 << str2 << endl;
 	
str1.clear();
cout << str1 << str2 << endl;
return 0;
}
	Try 1:
	text1
	Text2
	Text2text1 text1

Problems

1. Write a C++ program that reads two strings and then searches letter a in both of them.

Topic: Function

In C++ structure of function is:

data_type name_of_function (data_type parameter_name) {}

Parameters can be used from 0 to many as needed. The parameters are separated by commas (,). By parameter, function receives variable (arguments), which acts in the function as local variable.

C++ CODE	RESULT
#include <iostream>	Try 1:
using namespace std;	3 4
	Area : 12
int area(int a, int b)	3.1
{	-3.1
int ar;	
ar = a*b;	Try 2:
return ar;	12 -4
}	Area : -48
double f1(double x)	-3.009
{	3.009
return -1*x;	
}	
int main ()	
{	
int a1,a2; double d1,d2;	
cin>>a1>>a2;	
cout << "Area : " << area(a1,a2) << endl;	
cin>>d1;	
d2 = f1(d1);	
cout << d2;	
return 0;	
}	

Problems

1. Write a function max that returns maximum of two integer numbers.

2. Write a function sum that returns a sum of three double numbers.

Topic: Functions with no type “void”

In C++ programmer can write void type specifier in data_type for functions. Its structure is:

void name_of_function(data_type parameter_name) {}

Function with no type is used for example to print values of array, show message.

C++ CODE	RESULT
`#include <iostream>`	Try 1: Askar
`#include <string>`	Hi Askar
`using namespace std;`	13
	26
`void area(string str)`	
`{`	Try 2: Arman
`cout << "Hi " << str << endl;`	Hi Arman
`}`	25
`void show()`	50
`{`	
`int a;`	
`cin>>a;`	
`cout << a*2 << endl;`	
`}`	
`int main ()`	
`{`	
`string str; cin>>str;`	
`area(str);`	
`show();`	
`return 0;`	
`}`	

Problems

1. Write a program that reads 10 students name by loop then write a function print, which receives one string and prints this string with first capital letter.

Topic: Scope in function

In function variables are local variables for this function.

void function1(int x, int y) // x and y are created here

{

Here integers x, y and a are local integers.

int a=5; // a is created here

} // x, y and a are destroyed here

C++ CODE	RESULT
#include <iostream>	Try 1:
using namespace std;	2 3
	function x : 2
int area(int x, int y)	function y : 3
{	function a : 6
int a;	-------------
a = x*y;	function x : 3
cout << "function x : " << x << endl;	function y : 4
cout << "function y : " << y << endl;	function a : 12
cout << "function a : " << a << endl;	-------------
cout << "-------------" << endl;	main x : 6
return a;	main y : 12
}	main a : 2
int main ()	
{	
int a,b;	
cin>> a >> b;	
int x = area(a, b);	
int y = area(3, 4);	
cout << "main x : " << x << endl;	
cout << "main y : " << y << endl;	
cout << "main a : " << a << endl;	
return 0;	
}	

Topic: Forward declaration

A function prototype is a declaration of function without its implementation. Programmer declares function prototype at the top of the file to show to compiler that it will be used. This is called forward declaration.

If forward declaration of function is not used in code and function is written after main function then when programmer tries to use function after compiler gives error.

int main()
{
int a = sum(3,4);
…..
}
int sum(int x, int y)
{
return x+y;
}
In code in left there is a compilation error.
	int sum(int, int);
	int main()
	{
	int a = sum(3,4);
	…..
	}
	int sum(int x, int y)
	{
	return x+y;
	}
	No compilation error.

C++ CODE	RESULT
#include <iostream>	Try 1:
using namespace std;	5 6
int area(int, int); // forward declaration,	Area : 30
//try to write comment //	
// to understand and see compilation error	Try 2:
	12 2
int main ()	Area : 24
{	
int a,b;	
cin>> a >> b;	
int x = area(a, b);	
cout << "Area : " << x << endl;	
return 0;	
}	
int area(int x, int y)	
{	
return x*y;	
}	

Problems

1. Write functions area, perimeter of circle using forward declaration

Topic: External function

In C++ programmer can write a functions in other file and use them. To use external file in C++ code it must be included and have an extension .h. Also to include the structure must be like:

#include "name_of_file.h"

Note: there is a difference between <> and " " in C++. In <> standart class is included, in " " own external file is included.

C++ CODE	RESULT
sum.h	Try 1: 3 4 Area : 12 Try 2: 6 9 Area : 54

int area(int x, int y)
{
return x*y;
}	

main.cpp	

#include <iostream>
#include "sum.h" // here external file is included
using namespace std;
 	
int main ()
{
int a,b;
cin>> a >> b;
int x = area(a, b);
cout << "Area : " << x << endl;
return 0;
}	

Problems

1. Write two functions of finding maximum and minimum of 3 integers. Save both of them in test.h file and use in main.cpp

Topic: Namespace

When two identifiers (variable and/or function names) with the same name are introduced into the same scope a naming collision will result, and the compiler will produce an error because it does not have enough information to resolve the ambiguity.

A namespace defines an area of code in which all identifiers are guaranteed to be unique. By default, all variables and functions are defined in the global namespace. In order to help avoid issues where two independent pieces of code have naming collisions with each other when used together, C++ allows us to declare our own namespaces via the namespace keyword. Anything declared inside a user-defined namespace belongs to that namespace, not the global namespace.

C++ CODE	RESULT
#include <iostream>	Try 1:
using namespace std; namespace n1	5 3
{	namespace n1 : 8 namespace n2 : 2
int function(int x, int y)	
{ return x + y; }	Try 2:
}	-3 4
	namespace n1 : 1 namespace n2 : -7
namespace n2	
{	
int function(int x, int y)	
{ return x - y; }	
}	
int main ()	
{	
int a,b;	
cin>> a >> b;	
int x1 = n1::function(a, b);	
// namespace n1	
cout << "namespace n1 : " << x1 << endl;	
int x2 = n2::function(a, b);	
// namespace n2	
cout << "namespace n2 : " << x2 << endl;	
return 0;	
}	

Problems

1. Write two namespaces engineering and economy. In both namespaces there are variable price and function calculate, which returns 500 and 700 for engineering and economy respectively.

Topic: Passing arguments

There are 3 primary methods of passing arguments to functions: pass by value, pass by reference, and pass by address.

1. Pass by value. By default, arguments in C++ are passed by value. When arguments are passed by value, a copy of the argument is passed to the function.

2. Pass by reference. When arguments are passed by reference programmer can modify the value of argument. In pass by reference, address sign (&) is used. void function(int &y) // y is a reference variable

3. Pass by address. Pass by address means a passing the address of the argument variable rather than the argument variable itself. Because the argument is an address, the function parameter must be a pointer. The function can then dereference the pointer to access or change the value being pointed to. void function(int *y) // y is a pointer. function(&a); // in main programmer sends a pointer to variable not variable value

C++ CODE	RESULT
#include <iostream>	Try 1:
using namespace std;	3 5
void f1(int, int &);	3 5
void f2(int, int *);	3 5
int main ()	6 10
{	3 10
int a,b;	---------
cin>> a >> b;	3 10
cout << a << " " << b << endl;	3 10
f1(a,b);	9 30
cout << a << " " << b << endl;	3 30
cout << "--------------" << endl;	
cout << a << " " << b << endl;	
f2(a,&b);	
cout << a << " " << b << endl;	
return 0;	
}	
void f1(int a, int &b) {	
cout << a << " " << b << endl;	
a = a*2;	
b = b*2;	
cout << a << " " << b << endl; }	
void f2(int a, int *b) {	
cout << a << " " << *b << endl;	
a = a*3;	
*b = (*b) * 3;	
cout << a << " " << *b << endl; }	

Problems

1. Write two functions that swaps two integer numbers by reference, by address.

Topic: Class

A class is an expanded concept of a data structure: instead of holding only data, it can hold both data and functions.

An object is an instantiation of a class. In terms of variables, a class would be the type, and an object would be the variable.

Its structure is:

class name_of_class{

variables and functions

};

C++ CODE

```
#include <iostream>
#include <string>
using namespace std;
class person    // class description
{
public :
string name;   // variable
int age;          //  variable
void print()    // function
{
   cout << "Name : " << name << endl;
   cout << "Age : " << age << endl;
}
}; // here it is important to put ';' sign

int main ()
{
person student;           //object of class person
string n;
int a;
cin>> n >> a;
student.name = n;
student.age = a;
student.print();
return 0;
}
```

RESULT

```
Try 1: Arman 25
Name : Arman
Age : 25

Try 2: Andrey 19
Name : Andrey
Age : 19
```

Problems

1. Write a class date, which has 3 variables year, month and day. Make a function print which prints a date in format like “day of month, year”

2. Write a class book, which has 2 variables name, author. Make a function show which prints a book in format like “read name of author”

Topic: Access specifier

Access specifier is a declaration for data in class to give them rights of access. An access specifier is one of the following three keywords: private, public or protected.

- public members are accessible from anywhere where the object is visible.
- private members of a class are accessible only from within other members of the same class or from their friends.
- protected members are accessible from members of their same class and from their friends, but also from members of their derived classes (child classes).

Note: by default all data in class are private.

C++ CODE	RESULT
#include <iostream>	Try 1: Askar 25
#include <string>	Name : Askar
using namespace std;	Age : 25
class person {	
private : // variable name and age are private	Try 2: Arman 33
string name;	Name : Arman
int age;	Age : 33
public : // functions are public	
void set(string a, int b) {	
name = a;	
age = b; }	
void print() {	
cout << "Name : " << name << endl;	
cout << "Age : " << age << endl; }	
};	
int main () {	
person student;	
string n;	
int a;	
cin>> n >> a;	
student.set(n, a); // it works because of public	
//student.name = n; // without comment it is CE	
//student.age = a; // because they are private	
student.print(); // again public	
return 0;	
}	

Problems

1. Write a class student where two public functions setInfo and getInfo and 3 private variables name, GPA, course.

Topic: Access function

In class there can be a prototype of function declaration, but its implementation can be outside of it. In this case in outside programmer must use the operator of scope (::) to specify that a function that is a member of the class and not a regular global function.

In class programmer use the operator of scope (::, two colons) to define a member of a class from outside the class definition itself.

C++ CODE

```
#include <iostream>
#include <string>
using namespace std;
class student {
string name;
double GPA;
int course;
public :
void set(string, int, double); // only prototype
void print() {
cout << "Name : " << name << endl; cout << "Course : " <<
course << endl; cout << "GPA : " << GPA << endl; }
};
void student::set(string a, int b, double c) // implementation
{    name = a;
     course = b;
     GPA = c;  }

int main ()
{
student s1, s2;
string n;
int a;
double d;
cin>> n >> a >> d;
s1.set(n, a, d);
s1.print();
cout << "------------"<<endl;
cin>> n >> a >> d;
s2.set(n, a, d);
s2.print();
return 0; }
```

RESULT

```
Try 1:
Askar 3 3.8
Name : Askar
Course : 3
GPA : 3.8
-------------
Andrey 1 2.5
Name : Andrey
Course : 1
GPA : 2.5
```

Problems

1. Write a class Rectangle with private variable width and length, with functions setWidth, setLength and Area. Write all functions implementation outside of class.

Topic: Constructor

Constructor is a special function used to initialize variables in class. It is automatically called whenever a new object of this class is created. This constructor function must have the same name as the class, and cannot have any return type; not even void.

Its structure is:

name_of_class(data_type parameter)

{   C++ code }

C++ CODE	RESULT
#include <iostream>	Try 1:
#include <string>	Askar 2 3.5
using namespace std;	New student is created
class student {	Name : Askar
string name;	Course : 2
double GPA;	GPA : 3.5
int course;	
public :	Try 2:
student(string, int, double); // constructor	Olzhas 1 3.8
void print();	New student is created
};	Name : Olzhas
void student::print() {	Course : 1
cout << "Name : " << name << endl;	GPA : 3.8
cout << "Course : " << course << endl;	
cout << "GPA : " << GPA << endl; }	
student::student(string a, int b, double c) // implementation	
{	
cout << "New student is created " << endl;	
name = a; course = b; GPA = c; }	
int main () {	
string n;	
int a;	
double d;	
cin>> n >> a >> d;	
student s1(n, a, d);	
s1.print();	
return 0; }	

Problems

1. Write a class time1 wth constructor (int hour, int minute, int seconds).

2. Write a class notebook with constructor (string model, int year, string company)

Topic: Overloading constructors

In class there can one and more constructors with different types or number of parameters. This is called overloading of constructor. All of them must have the same name, but the compiler will call only one whose parameters match the arguments.

C++ CODE

```cpp
#include <iostream>
using namespace std;
class rectangle {
int width, height;
public :
rectangle(); // with no parameters
rectangle(int); // with parameter width (square)
rectangle(int, int); // with parameters width and height
int area();
};
rectangle::rectangle() {
width = height = 0;
cout << "Without any parameter " << endl; }
rectangle::rectangle(int w) {
width = w;
height = w;
cout << "Square is created " << endl; }
rectangle::rectangle(int w, int h) {
width = w;
height = h;
cout << "Rectangle is created " << endl; }
int rectangle::area() {
return width*height;  }

int main ()  {
int a,b;
cin >> a >> b;
rectangle r1;    // constructor1
rectangle r2(a);  // constructor2
rectangle r3(a, b); // constructor3

cout << r2.area() << " " << r3.area();
return 0; }
```

RESULT

```
Try 1:
3 5
Without any parameter
Square is created
Rectangle is created
9 15

Try 2:
4 7
Without any parameter
Square is created
Rectangle is created
16 28
```

Problems

1. Write a class cellPhone with variables manufacture, model, price and color. Write 2 overloading constructors for this class. Write a function changePrice and changeColor.

1. Write a class book with variables name, author, pages and ISBN. Write 3 overloading constructors for this class. Write a function changeName and print.

Topic: Default constructor

If there is not declaraton of constructor in a class definition, the compiler assumes that the class has a default constructor with no arguments. So programmer can declare objects of class by simply declaring them without any arguments. But as soon as programmer declares own constructor for a class, the compiler no longer provides an implicit default constructor. And programmer has to declare all objects of that class according to the constructor prototypes.

class example1
{
public :
int a;
void setA(int b) { a= b; }
};
	class example2
	{
	public :
	int a;
	example2(int b) { a= b; }
	};

without constructor, so we can write like example1 d; and cannot write like example1 d(5);	with constructor, so we can write like example2 d(5); and cannot write like example2 d;

C++ CODE

```
#include <iostream>
using namespace std;
class example1 {
public :
int a;
void setA(int b) { a= b; }
};

class example2 {
public :
int a;
example2(int b) { a= b; }
};

int main ()
{
int x;
cin >> x; example1 r1; r1.setA(x);
example2 r2(x);
cout << r1.a << " " << r2.a;
return 0;
}
```

RESULT

```
Try 1:
3
3 3

Try 2:
4
4 4
```

Problems

1. Write class engineering class with constructor and economy class without constructor with function print.Write a main class to use them.

Topic: Destructor

The destructor is opposite of constructor. Normally it is automatically called when an object is destroyed, either because its scope of existence has finished or because it is an object dynamically assigned and it is released using the operator delete. But programmer can also make his destructor.

Its structure is:

~name_of_class () {

C++ Code

}

Note: in class there can be only one destructor.

C++ CODE	RESULT
#include <iostream>	Try 1:
#include <string>	3 5
using namespace std;	3 5
class book {	3 5
string *name;	6 10
int *ISBN;	3 10
public :	---------
book(string n, int i); // constructor	3 10
~book(); // destructor	3 10
};	9 30
book::book(string n, int i) {	3 30
name = new string(n); ISBN = new int;	
*ISBN = i;	
cout << "object " << n << endl; }	
book::~book() {	
cout << "Object is destructed" << endl;	
delete ISBN;	
delete name; }	
int main () {	
string s;	
int x;	
cin >> s >> x;	
{ book b1(s, x); }	
// here destructor is working book b2("AAA",x);	
return 0;	
} // here destructor is working	

Problems

1. Write a class student with destructor.

Topic: External class

Programmer can define and implement a class to use to other sources.

Mostly in (.h) header file there is a class definition. And in (.cpp) there is an implementation of this class.

C++ CODE

example.h

```
class example
{
int i;
double d;
public :
example();
void set(int, double);
void print();
};
```

example.cpp

```
#include "example.h"
#include <iostream>
using namespace std;

example::example() { }

void example::set(int a, double q)  {
i = a;
d = q;  }

void example::print() {
cout << "Integer " << i << endl;
cout << "Double " << d << endl;  }
```

main.cpp

```
#include <iostream>
#include "example.h"
using namespace std;

int main()
{
int x;
double d;
cin >> x >> d;
example s1;
s1.set(x, d);
s1.print();
return 0; }
```

RESULT

```
Try 1:
2 3.1
Integer 3
Double 3.1

Try 2:
4 4.11
Integer 4
Double 4.11
```

Problems

1. Write a C++ class book with header and implementation class. Write a main class for testing.

Topic: Pointer to class

In C++ programmer can make a pointer to class. Its structure is:

name_of_class *name_of_object;

To call functions there are two ways:

1 ¬ (*name_of_object).method();

2 – name_of_object¬>method();

C++ CODE

```
#include <iostream>
#include <string>
using namespace std;
class Book {
string name, author;
public :
Book();
void setBook(string, string);
void print();
};
Book::Book() {
name = "Unknown";
author = "Unknown"; }
void Book::setBook(string n, string a) {
name = n;
author = a; }
void Book::print() {
cout << "Name " << name << endl;
cout << "Author " << author << endl;}

int main ()
{
Book r1;
r1.print();
string a,b;
cin >> a >> b;
Book *r2 = new Book[2];
r2->setBook("Abay joli", "Auezov");
r2[0].print();
r2[1].setBook(a,b);
r2[1].print();

return 0;  }
```

RESULT

```
Try 1:
Name Unknown
Author Unknown
Abay
Kozimnin Karasi
Name Abay joli
Author Auezov
Name Abay
Author Kozimnin

Try 2:
Name Unknown
Author Unknown
Okulik
Altynsarin
Name Abay joli
Author Auezov
Name Okulik
Author Altynsarin
```

Problems

1. Write a C++ class mobile phone with pointer to class.

Topic: Pointer "this"

The keyword this identifies a special type of pointer. It is used to store the address of called object.

When programmer creates an object named x of class A, and class A has a nonstatic member function f(). If programmer calls the function x.f(), the keyword "this" in the body of f() stores the address of x.

Note: Programmer cannot declare "this" pointer or make assignments to it.

C++ CODE	RESULT
#include <iostream>	Try 1:
using namespace std;	2
class calc {	2
int a;	0
public:	15
calc();	
void set(int);	Try 2:
int get();	4
calc& plus(int);	4
};	0
calc::calc() {	15
a = 0; }	
void calc::set(int x) {	
a = x; }	
int calc::get(){	
return a; }	
calc& calc::plus(int x) {	
a += x;	
return *this; }	
int main () {	
calc c1;	
calc c2;	
int x;	
cin >> x;	
c1.set(x);	
cout << c1.get() << endl;	
cout << c2.get() << endl;	
c2.plus(10).plus(5);	
cout << c2.get() << endl;	
return 0; }	

Problem

1. Write a C++ class number with object n1 and pointer to n1. Make a function check, which checks if the pointer is same with object.

Topic: Overloading operators

To overload an operator in order to use it with classes programmer declares operator functions, which are regular functions whose names are the operator keyword followed by the operator sign. Its structure is:

data_type operator sign(parameters)

{ C++ CODE }

C++ CODE	RESULT
#include <iostream>	Try 1:
using namespace std;	1 2
	3 3
class number {	
public:	Try 2:
int x;	3 7
number () {};	10 10
number (int);	
number operator+(number);	
};	
number::number(int a) {	
x = a; }	
number number::operator+(number n1) {	
number temp;	
temp.x = x + n1.x;	
return temp;	
// return number(x+n1.x);	
}	
int main () {	
int a1, a2;	
cin >> a1 >> a2;	
number a(a1);	
number b(a2);	
number c, d;	
c = a + b;	
d = a.operator+(b);	
cout << c.x << " " <<d.x <<endl;	
return 0; }	

Problems

1. Write a class number that subtracts two numbers.

Topic: Overloading unary operator

Whenever an unary operator is used, it works with one operand, therefore with the user defined data types, the operand becomes the caller and hence no arguments are required.

C++ CODE	RESULT

#include <iostream>
using namespace std;
 	
class number {
public:
int x;
number () {};
number (int);
number operator ++();
number operator ++ (int);
};
number::number(int a) {
x = a; }
number number::operator++()
{   return number(++x);   }
number number::operator++(int)
{ return number(x++);   }
 	
int main () {
int a1, a2;
cin >> a1 >> a2;
number a(a1);
number b(a2);
a++;
++b;
cout << a.x << " " <<b.x <<endl;
return 0; }
	Try 1:
	1 2
	2 3
	
	Try 2:
	34 45
	35 46

Problems

1. Write a class number with decrement operator.

Topic: Overloading binary operator

Whenever a binary operator is used ¬ it works with two operands, therefore with the user defined data types ¬ the first operand becomes the operator overloaded function caller and the second is passed as an argument. This results in compulsion of receiving one argument in overloading of the binary operators.

C++ CODE

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

class number {
int x;
public:
number () {};
number (int);
bool isEqual(number);
};
number::number(int a) {
x = a;   }
bool number::operator==(number n1) {
return (x == n1.x);  }

int main () {
int a1, a2;
cin >> a1 >> a2;
number a(a1);
number b(a2);
if (a==b) cout << "Equal" <<endl;
else cout << "Not equal";
return 0;
}
```

RESULT

Try 1:
2 3
Not equal

Try 2:
3 3
Equal

Problems

1. Write a class number with checking greater, less and not equal

Topic: Overloading I/O operators

In C++ there are overloading functions of cin and cout. This means that programmer can make class function to read data as he needs.

C++ CODE	RESULT
`#include <iostream>`	Try 1:
`using namespace std;`	3 5
	x : 3 y : 5
`class Cnumber {`	4 5
`int x, y;`	x : 4 y : 5
`public:`	
`Cnumber () {};`	Try 2:
`Cnumber (int, int);`	5 5
`friend istream& operator>> (istream &in, Cnumber &t);`	x : 5 y : 5
`friend ostream& operator<< (ostream &out, Cnumber &t);`	12 13
`};`	x : 12 y : 13
`Cnumber::Cnumber(int a, int b) {`	
`x = a;`	
`y = b; }`	
`istream& operator>>(istream &in, Cnumber &t) {`	
`in >> t.x; in >> t.y;`	
`return in; }`	
`ostream& operator<< (ostream &out, Cnumber &t) {`	
`out << " x : " << t.x << " y : " << t.y << endl;`	
`return out; }`	
`int main () {`	
`Cnumber a1, a2, a3;`	
`cin >> a1;`	
`cout << a1;`	
`cin >> a2;`	
`cout << a2;`	
`return 0; }`	

Problems

1. Wrte a class number with overloading I/O operators

2. Write a class time1 wth variable int hour, int minute, int seconds. Make overloading I/O operators.

Topic: Friend function

A friend function is a function that can access the private members of a class as though it were a member of that class. In all other regards, the friend function is just like a normal function.

C++ CODE	RESULT

#include <iostream>
using namespace std;
 	
class number {
int x;
public:
number () {};
number (int);
friend number operator+(number, number);
void print() { cout << x;}
};
number::number(int a) {
x = a; }
number operator+(number t1, number t2) {
number temp;
temp.x = t1.x + t2.x;
return temp; }
 	
int main () {
int a1, a2;
cin >> a1;
number a(a1);
cin >> a2;
number b(a2);
 	
number c;
c = a + b;
c.print();
return 0; }
	Try 1:
	3 5
	8
	
	Try 2:
	4 7
	11

Problems

1. Write a friend operator- for class number with two different arguments:

   1 with object

   2 with integer.

<u>Topic: Friend class</u>

A friend class defines a class as friend of another one, granting that first class access to the protected and private members of the second one.

C++ CODE

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

class CSquare;
class CRectangle {
int width, height; public:
int area () { return (width * height); }
void convert (CSquare a);
};
void CRectangle::convert (CSquare a) {
width = a.side;
height = a.side;
}

class CSquare {
private: int side; public:
void set_side (int a)  {side=a;}
friend class CRectangle;
};

int main ()
{
CSquare sqr;
CRectangle rect;
int a;
cin >> a;
sqr.set_side(a);
rect.convert(sqr);
cout << rect.area();
return 0; }
```

RESULT

Try 1:
3
9

Try 2:
4
16

Topic: Inheritance

Inheritance is a way of creating classes which are derived from other classes, so that they automatically include some of its "parent's" members, plus its own. The class from which inherited is called the parent or base, and the class to which inheriting is called the child or derived object.

C++ CODE

```cpp
#include <iostream>
using namespace std;
class person{
public:
string name;
int age;
void setName(string s) { name = s; }
void setAge(int i){ age = i; }
string getName() { return name; }
int getAge() { return age; }
void print() {
cout << "name:" << name << " age:" << age << endl;}
};
class student:public person{
public:
double GPA;
int course;
void setGPA(double d) { GPA = d; }
void setCourse(int i) { course = i; }
void info() {
    print();
    cout << "GPA:" << GPA<< " Course:" << course;  }
};
int main () {
student s1;
string name;
cin >> name;
s1.setName(name);
int age;
cin >> age;
s1.setAge(age);
s1.print();
double g;
int c;
cin >> g >> c;
s1.setGPA(g);
s1.setCourse(c);
s1.info();
return 0; }
```

RESULT

```
Try 1:
Askar
21
Name:Askar age:21
3.4
2
Name:Askar age:21
GPA:3.4 Course:2

Try 2:
Arman
19
Name:Arman age:19
2.6
1
Name:Arman age:19
GPA:2.6 Course:1
```

Problems

1. Write a class shape with variables width and height. Write a class rectangle child of shape with function area.

Topic: Protected member

The protected access specifier is similar to private. Its only difference occurs in fact with inheritance. When a class inherits from another one, the members of the derived class can access the protected members inherited from the base class, but not its private members.

C++ CODE	RESULT
#include <iostream>	Try 1:
using namespace std;	Askar Askarov
class person{	2.3
private:	1
string surname;	Student name:Askar
protected: string name; int age;	GPA:2.3 Course:1
public:	
void setName(string s) { name = s; }	Try 2:
void setAge(int i){ age = i; }	Arman Amangeldi
void setSurname(string s) { surname = s; }	3.6
void print() {	3
cout << "name:" << name << " age:" << age << endl;}	Student name:Arman
};	GPA:3.6 Course:3
class student:public person{	
public:	
double GPA;	
int course;	
void setGPA(double d) { GPA = d; }	
void setCourse(int i) { course = i; }	
void info() {	
cout << "Student name:" << name << endl;	
// cout << "Student surname:" << surname << endl;	
// line up cannot be used because surname is private;	
cout << "GPA:" << GPA<< " Course:" << course << endl; }	
};	
int main () {	
student s1;	
string name, surname;	
cin >> name;	
s1.setName(name);	
cin >> surname;	
s1.setSurname(surname);	
double g;	
int c;	
cin >> g >> c;	
s1.setGPA(g);	
s1.setCourse(c);	
s1.info();	
return 0; }	

Problems

1. Write a class point with protected int x, y. Write a class line a child class of point.

Topic: Inheritance relationship

There are three different ways inheritance relationship: public, private, and protected.

class Child: public Parent {}; All members keep their original access specifications. Private members stay private, protected members stay protected, and public members stay public.

class Child: private Parent {}; private members stay private, and protected and public members become private.

class Child: protected Parent

{}; the public and protected members become protected, and private members stay private class Child: Parent {}; By default private is used.

C++ CODE	RESULT

#include <iostream>
using namespace std;
class shape{
private:
int Z;
protected:
int width, height;
public:
void set(int a, int b) {
width = a;
height = b; }
};
class rectangle:private shape{
public:
int getArea() { return width*height; }
};
class square:public rectangle{
public:
int getArea() { return width*height; }
};
 	
int main () {
rectangle r;
square s;
return 0; }
	Try 1:
	Compilation error
	
	Class square tries to use shape's width and height which are private for it.

Problems

1. Try to change code to work properly

Topic: Initialization of derived classes

In principle, a derived class inherits every member of a base class except:

- its constructor and its destructor
- its operator=() members
- its friends

To overload constructor of base class a new derived object is created with structure like:

derived_constructor_name (parameters) : base_constructor_name (parameters)

{ C++ code }

C++ CODE

```cpp
#include <iostream>
using namespace std;
class A {
public:
A(int n) {  cout << "A: " << n << endl; }
};
class B: public A
{
public:
B(int n, double d) : A(n) {
cout << "B: " << d << endl; }
};
class C: public B {
public:
C(int n, double d, char ch) : B(n, d) {
cout << "C: " << ch << endl;  }
};

int main()
{
int i1;
double i2;
char i3;
cin >> i1 >> i2 >> i3;
C c1(i1, i2, i3);
return 0;
}
```

RESULT

```
Try 1:
5
3.4
Z
A: 5
B: 3.4
C: Z

Try 2:
9
4.1 q
A: 9
B: 4.1
C: q
```

Topic: Multiple inheritance

Multiple inheritances enable a derived class to inherit members from more than one parent. Its structure is:

class name_of_derived_class : public parent1_class, public parent2_class e.t.c

{ C++ code };

C++ CODE	RESULT

#include <iostream>
using namespace std;
class person {
protected:
string name;
public:
void setName(string n) { name = n; }
};
class mobilePhone {
protected:
string phone;
public:
void setPhone(string n) { phone = n;}
};
class student: public person, public mobilePhone {
int course;
public:
void info() {
cout << "Student name : " << name << endl;
cout << "Phone number : " << phone << endl;
cout << "Course : " << course << endl;
}
void setCourse(int i) { course = i; }
};
 	
int main() {
string n, p;
student s1;
cin >> n;
s1.setName(n);
cin >> p;
s1.setPhone(p);
int i;
cin >> i;
s1.setCourse(i);
s1.info();
return 0; }
	Try 1:
	Arman
	87019999999
	2
	Student name : Arman
	Phone number :
	870199999999
	Course : 2
	
	Try 2: Askar
	2738800
	1
	Student name : Askar
	Phone number : 2738800
	Course : 1

Problems

1. Write a class child, which has 2 parent classes: father and mother.

Reference

1. www.cplusplus.com

2. www.learncpp.com

3. http://www.codingunit.com

4. http://www.mochima.com/tutorials