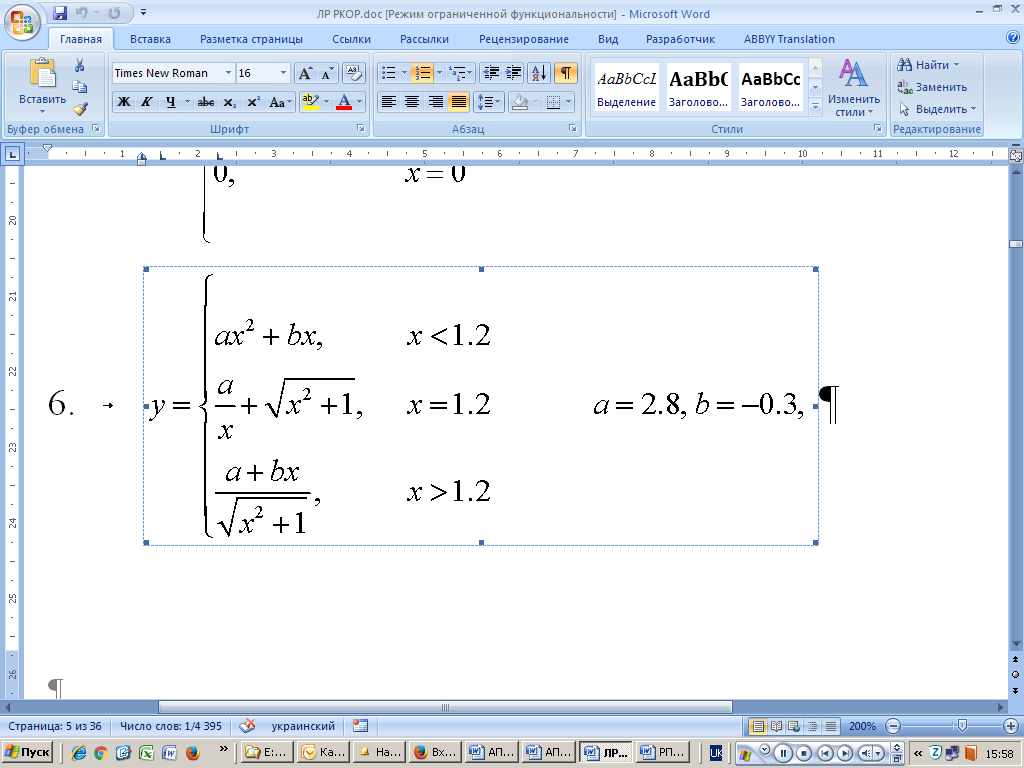
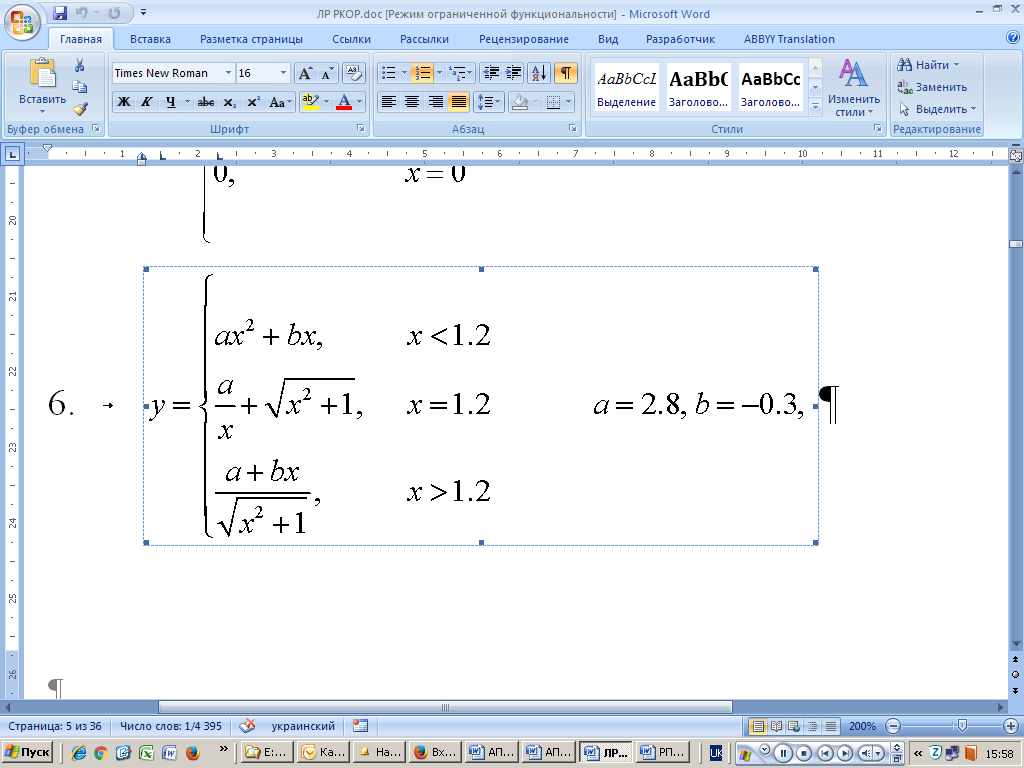
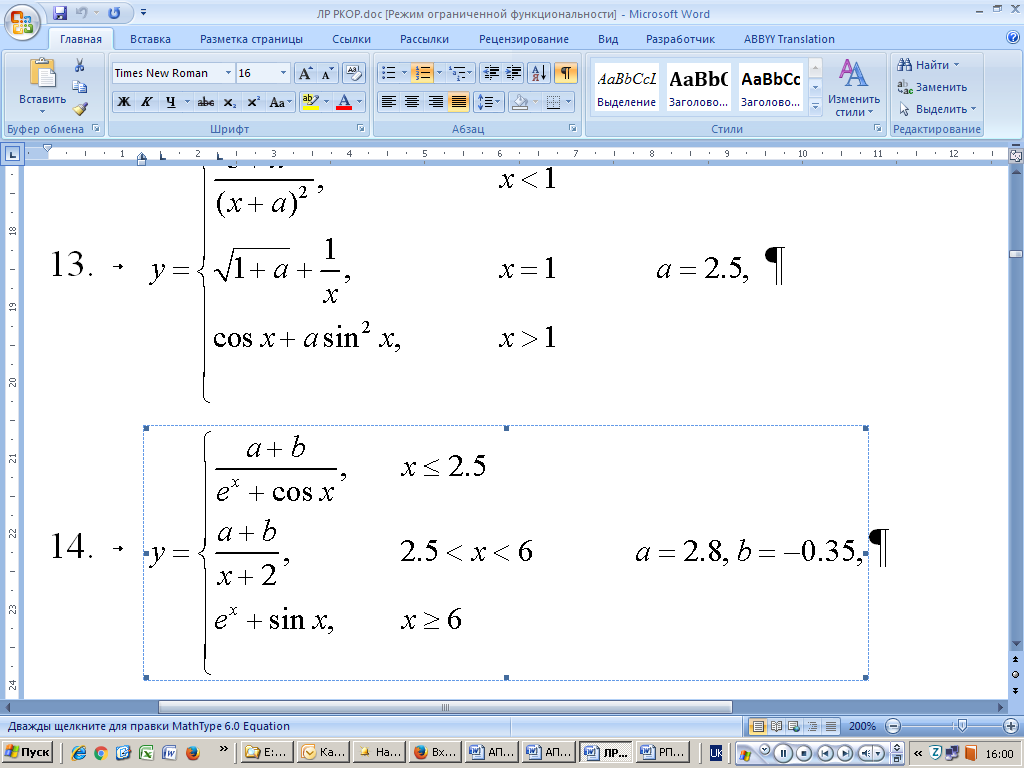
**Практична робота №4. Використання математичних функцій, введення – виведення даних.**

**Завдання.**

Запустити в Інтернет-браузері он-лайн компілятор C++Shell за посиланням <http://cpp.sh/> або використати інший доступний вам компілятор С++

1. Написати на мові С++ програму введення даних для розрахунку формул, наданих викладачем, а також виведення результатів. Використати потокове введення та виведення даних. Значення a, b задати константами.

Примітка. Значення a, b задати константами. Перед введенням даних вивести на консоль назву змінної, дані якої будуть вводитися. Результат виводити по кожній формулі окремо, але одним рядком, вказуючи номер формули.

**  **

2. Після ознайомлення з теоретичною частиною написати оператори форматованого введення та виведення даних за завданням наданим викладачем.

Модифікуйте Приклад 1, замінивши присвоєння змінним, які визначають вік, зарплату, доплату, введенням цих даних з виведенням відповідної назви. Розрахуйте відсоток доплати від зарплати. Виведення даних модифікуйте, використавши **setw(w)** та **setprecision(d)**.

3. Пп. 1-2 можна об’єднати в одну програму. Першим рядком програми повинен бути коментар з прізвищем студента на назвою роботи (наприклад, // Батс ПР 4).

Результати у вигляді текстового файлу надсилати на електронну адресу викладача [**t.i.lumpova@gmail.com**](mailto:t.i.lumpova@gmail.com)

Файл повинен мати назву в такому форматі:

**АР<Номер групи><Номер лекції / практичної / лабораторної [літера позначення типу роботи L – лекція, P – практична, R – лабораторна]<Прізвище англійською>**. Наприклад, **АРPTBD-2104P**buts.doc.

**Тему в заголовку листа записати**

**АР <Номер групи>-><Номер лекції / практичної / лабораторної [літера позначення типу роботи L – лекція, P – практична, R – лабораторна]<Прізвище англійською>**

Іншим рішенням є надсилання поштою посилання на текст програми за URL адреси, яку надає C++Shell, вказавши в темі листа, номер групи прізвище студента та номер ПР.

**Строк виконання цієї роботи і тесту - 17.02.2021**

Всі запитання, що виникнуть, надсилайте на електронну адресу викладача, тему в заголовку листа записати

**АР<Номер групи>-Запитання-<Прізвище англійською>**.

**Теоретична частина**

Директиви підключення ресурсів. Служать для підключення до тексту програми текстів *заголовних файлів* (мають стандартне розширення .h). ***Заголовний файл*** – це файл, що містить описи деяких функцій та оголошення типів даних і констант, які використовуються цими функціями. Список заголовних файлів визначається стандартом мови.

Існують два способи підключення директиви #include:

1. # include <ім'я> - підключення стандартних заголовних файлів;
2. # include "ім'я" - підключення заголовних файлів користувача.

Різниця між ними полягає в тому, де препроцесор буде шукати файли-ресурси, які необхідно підключити для повної підготовки коду програми до компіляції: якщо ім’я файла взяте в кутові дужки <…>, то пошук буде вестись в наперед визначених каталогах; якщо ім’я взяте в лапки, то препроцесор шукає його в поточному каталозі, де міститься і основний файл компілювання.

Слід зазначити, що багато стандартних заголовних файлів має порожнє розширення, для решти - традиційно використовують розширення h. Наприклад,

#include <iostream> *// підключення засобів потокового введення-виведення*

#include < stdio.h> *// підключення засобів форматованого введення-виведення*

Основні заголовні файли, які входять до складу стандартної бібліотеки, подані в табл.1.

Таблиця 1. Основні заголовні файли стандартної бібліотеки С/С++

| *Заголовний файл* | *Вміст* |
| --- | --- |
| <iosream.h> | прототипи для функцій стандартних вхідних і вихідних потоків |
| <iomanip.h> | прототипи функцій для маніпуляторів, що дозволяють форматувати потоки даних |
| <math.h> | прототипи функцій математичної бібліотеки |
| <stddef.h> | загальні визначення типів, що використовуються в С/С++ для виконання деяких обчислень |
| <stdio.h> | прототипи для функцій вводу/виводу стандартної бібліотеки і інформація, що ними використовується |
| <stdlib.h> | прототипи функцій для перетворення чисел в текст і тексту в числа, прототипи функцій розміщення в пам’яті, генерації випадкових чисел і інших сервісних функцій |
| <time.h> | прототипи функцій і типи для функцій управління часом і датою |

Розглянемо найпростіший приклад програми на C/*С++* :

**#include <iostream> *// директива препроцесора***

**using namespace std; *// простір імен***

**int main() *// головна функція***

**{ cout << "Hello,world\n"; *// виведення повідомлення на екран***

**system("pause"); *// пауза***

**return 0; *// повернення результату (вихід із функції)***

**}**

Рядок #include (включити) означає, що препроцесор перед компіляцією програми має включити в неї вміст заголовного файлу iostream.h. У цьому файлі оголошено засоби потокового введення та виведення. Без включення цього файлу ім’я cout буде невизначеним і компілятор повідомить про цю помилку.

У другому рядку розташовано інструкцію компілятору «використати простір імен std». Не пояснюючи значення слів «простір імен», скажемо лише, що простір імен std є стандартним. У сучасних системах програмування мовою С++ у ньому описано всі бібліотечні засоби «останнього покоління». Проте не всі компілятори розуміють цю інструкцію, тому її можна писати не завжди.

Частина програми

**int main()**

**{ ...**

**}**

визначає головну функцію main.

Операція << ("помістити в") пише свій перший аргумент у другий (рядок "Hello,world\n" у стандартний потік виведення cout). У даному рядку символ “\n“ є управляючим символом переходу на новий рядок.

За інструкцією system("pause"); виконання програми призупиняється і на екрані з’являється повідомлення, що треба натиснути будь-яку клавішу. Після її натискання програма завершується. Завдяки використанню управляючого символу “\n“, повідомлення виводиться в новому рядку. Якби його не було, повідомлення з’являлося б відразу після слів Hello, world!.

# Потокове введення-виведення Приклади

***Приклад 1.*** Написати програму, що містить організацію виведення даних, пояснювальні повідомлення, а також символи переведення рядка.

**#include <iostream>**

**using namespace std;**

**int main()**

**{**

**int wozrast = 20;**

**int doplata = 2;**

**float zarplata = 309.75;**

**float prozent = 8.5;**

**cout << " Verification of source data \n";**

**cout << " Age"<< '\t' <<"additional payment "<< '\t' << "salary "<< '\t' << "percent:\n";**

**cout << wozrast << '\t' << doplata << '\t' << zarplata << '\t' << prozent << '\n';**

**}**

В останніх двох операціях виведення програми використано символи табуляції.

Для додаткового керування даними, що виводяться, використовують маніпулятори **setw(w)** та **setprecision(d)**. Маніпулятор **setw(w)**призначений для зазначення довжини поля, що виділяється для виведення даних (**w** — кількість позицій). Маніпулятор **setprecision(d)** визначає кількість позицій у дробовій частині дійсних чисел.

Маніпулятори змінюють вигляд деяких змінних в об’єкті **cout**, що у потоці розташовані за ними. Ці маніпулятори називають *прапорцями стану*. Коли об’єкт посилає дані на екран, він перевіряє прапорці, щоб довідатися, як виконати завдання, наприклад, запис:

**cout << 456 << 789 << 123;**

призводить до виведення значення у вигляді: **456789123**, що ускладнює визначення групи значень.

***Приклад 2.*** Написати програму, використовуючи маніпулятор **setw()**.

**#include <iostream.h>**

**#include <iomanip.h>**

**#include <conio.h>**

**main ( )**

**{**

**cout << 456 << 789 << 123 << endl;**

**cout << setw(5) << 456 << setw(5) << 789 << setw(5) << 123 << endl;**

**cout << setw(7) << 456 << setw(7) << 789 << setw(7) << 123 << endl;**

**getch ();**

**}**

Результати виконання програми:  
**456789123**   
**456     789     123**   
**456       789       123**

У цьому прикладі з’явився новий заголовний файл **iomanip.h**, що дозволяє застосовувати функції маніпуляторів. При використанні функції **setw()** число вирівнюється вправо в межах заданої ширини поля виведення. Якщо ширина недостатня, то вказане значення ігнорується.

Функція **setprecision(2)** повідомляє про те, що число з плаваючою крапкою виводиться з двома знаками після крапки з округленням дробової частини, наприклад, при виконанні операції

**cout** **<< setw(7) << setprecision(2) << 123.456789;**

буде отримано такий результат: **123.46.**

Функції **cout.width(w)** та **cout.precision(d)**, які потребують підключення тільки заголовного файла **iostream.h**, виконують дії, подібні тим, що і функції **setw(w)** та **setprecision(d)**.

Операція введення використовує ті ж самі маніпулятори, що й операція виведення.

***Приклад 3.*** Написати програму обчислення податку на продаж.

**#include <iostream.h>**

**#include <iomanip.h>**

**#include <conio.h>**

**main ( )**

**{ float prod\_sum;** *// prod\_sum — сума продаж*

**float nalog;**

cout << **" Enter the amount of sales ";**

**cin >> prod\_sum;**

*//............... обчислення податку на продаж*

**nalog = prod\_sum\* 0.7;**

**cout << " " << setprecision(2) << prod\_sum;**

**cout << " " << setprecision(2) << nalog << "\n";**

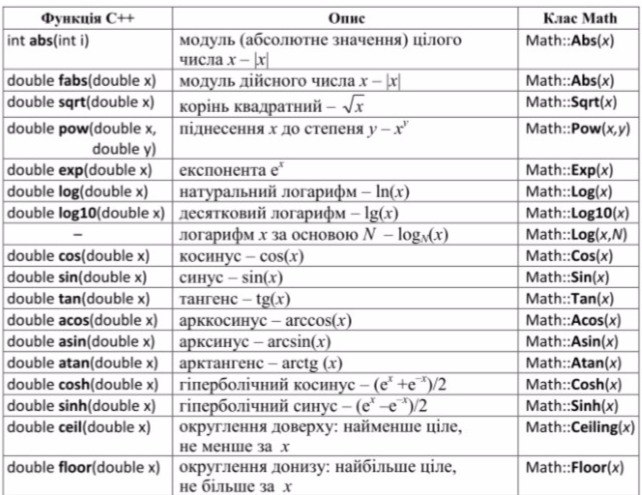
**getch();** *// затримка екрана*

**return 0;**

**}**

Внаслідок того, що у першому операторі **cout** відсутня інструкція переведення рядка, відповідь користувача на підказку (тобто введене значення змінної **prod\_sum**) з’явиться відразу праворуч за самою підказкою.

**Основні математичні функції C++ та класу Math**

****

**Правила запису математичних виразів**

* Кількість відкритих і закритих дужок у виразах повинна бути однаковою.
* Усі елементи виразів (дроби, показник степеня, індекси) записують у горизонтальному рядку.
* Вирази можна записувати у декількох рядках. "Розривати” вирази можна, наприклад, після символу арифметичної операції. Власне символ дублювати не потрібно.
* Операції ++ (інкремент) та -- (декремент) є унарними, тобто мають лише один операнд.
* Операція ++ додає одиницю до операнда, операція -- віднімає одиницю від операнда.

Обчислення в арифметичних виразах виконуються зліва направо згідно з  таким пріоритетом операцій:

1. стандартні функції, ++, --;

2. множення (\*), ділення (/), остача від ділення (%);

3. додавання (+) та віднімання (–).

Для здобуття правильного результату слід дотримуватися таких правил записування арифметичних виразів в операторах С++:

* кожна команда (інструкція) має завершуватись крапкою з комою (;)
* мова С++ є чутлива до регістру, тобто х та Х – це дві різні змінні;
* аргумент функції завжди записують у круглих дужках;
* знаки множення не можна пропускати (3ab –> 3\*a\*b);
* якщо знаменник або чисельник має операції (+, −, \*, /), то його слід записувати у круглих дужках;
* для записування раціональних дробів, у чисельнику або знаменнику яких є числові константи, хоча б одну з цих констант слід записати як дійсне число із зазначенням десяткової крапки, наприклад, 2⁄ 𝑘 записують як 2.0/k;
* радикали (тобто корінь кубічний і вище) замінюють на дробові степені, наприклад, кубічний корінь (𝑥 + 1) записують як  pow(x+1, 1/3.0);
* слід враховувати правила зведення типів, оскільки в арифметичних виразах можуть брати участь різнотипні дані та відбувається зведення типів.

**Приклади запису математичних виразів**

***Контрольні запитання***

1. Для чого слугують заголовні файли?
2. Як компілятор шукає розташування заголовних файлів?
3. Які маніпулятори використовуються для завдання довжини поля, дробової частини числа?
4. Який заголовний файл використовується для маніпуляторів завдання довжини поля, дробової частини числа?
5. Як забезпечується організація введення даних на С++?
6. Як здійснюється виведення результатів на С++?
7. Як реалізується потокове введення даних «**сіn>>**»? Наведiть приклади.
8. Охарактеризуйте основні аспекти використання потокового виведення даних «**соut<<**».
9. Що таке форматоване введення-виведення даних?
10. Поясніть правила застосування функції **scanf**.
11. Як здійснює роботу функція **printf**?