

23 варіант

1. Охарактеризувати поняття явне та неявне перетворення типів.

Є 2 основних способи конвертації типів:

- Неявна конвертація типів, коли компілятор автоматично конвертує один фундаментальний тип даних в інший. Виконується коли ми передаємо в змінну значення не її типу (наприклад число з плаваючою комою в `int` = компілятор неявно заокруглить до більшого або меншого).

Неявна конвертація відбувається одним з двох способів:
числове розширення;
числова конверсія.

Коли значення з одного типу даних конвертується в інший тип даних, який є більшим (за розміром і по діапазону значень), то це називається числовим розширенням. Наприклад, тип `int` може бути розширений в тип `long`.



```
long l(65); // розширюємо значення типу int (65) в тип long
double d(0.11f); // розширюємо значення типу float (0.11) в тип double
```

Натомість, числова конверсія – конвертація значення з більшого типу даних в аналогічний, але менший тип.

```
double d = 4; // конвертуємо 4 (тип int) в double
short s = 3; // конвертуємо 3 (тип int) в short
```

- Явна конвертація типів, коли розробник використовує один з операторів конвертації для виконання конвертації об'єкта одного типу даних в інший.

Перевагою явної конвертації є значне спрощення читання коду – замість уважно придивлятися до правильності переданих значень, розробник одразу бачить який тип даних використовується у певній операції.

Наприклад виконання цього коду буде 3:	результат	А цього – необхідний нам дробовий результат.
<pre>13 int main() 14 { 15 int i1 = 11; 16 int i2 = 3; 17 float x = i1 / i2; 18 19 cout<<x; 20 } 21</pre>		<pre>13 int main() 14 { 15 int i1 = 11; 16 int i2 = 3; 17 float x = float(i1) / i2; 18 19 cout<<x; 20 } 21</pre> 

Такий тип явного перетворення називається C-style.

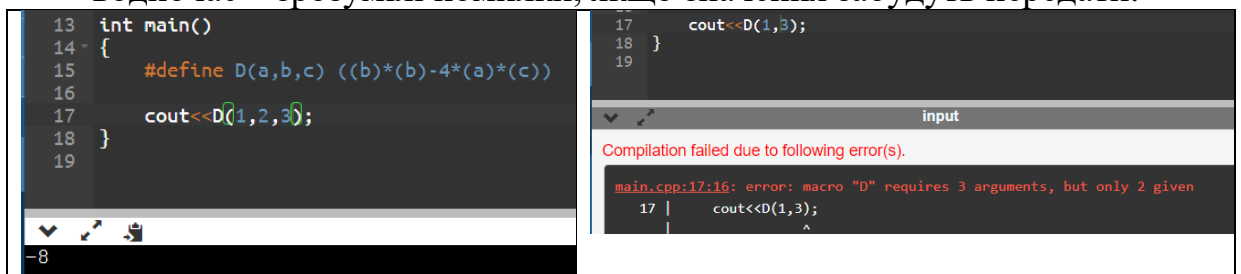
2. Охарактеризувати директиви препроцесора. Навести приклади.

Препроцесор – це програма, яка опрацьовує директиви. Директиви препроцесора – це команди компілятора, котрі виконуються на початку компіляції програми. У мові C++ директиви починаються з символу # а опісля містять необхідну директиву без пробілу. Після директиви НЕ ставиться крапка з комою.

Приклади:

`#include` – означає, що до програми необхідно приєднати програмний код із зазначеного після неї файлу. Файли, які приєднують директивою `#include` називаються файлами заголовків (header-файли, також відомі як модулі або бібліотеки). У таких файлах зазвичай оголошують сталі та змінні, заголовки функцій котрі часто використовуватимуться у файлі.

`#define`, `#undef` – записати константу і переписати на звичайну змінну – котрій при зверненні можна змінювати значення. Перевагою директиви `#define` є краще сприйняття коду – швидше та приємніше читання, і водночас – зрозумілі помилки, якщо значення забудуть передати.



```
13 int main()
14 {
15     #define D(a,b,c) ((b)*(b)-4*(a)*(c))
16
17     cout<<D(1,2,3);
18 }
19
```

input

Compilation failed due to following error(s).

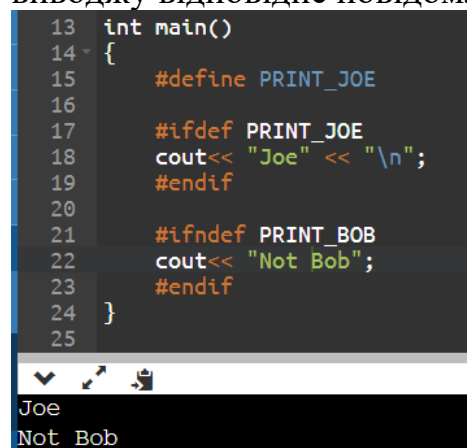
main.cpp:17:16: error: macro "D" requires 3 arguments, but only 2 given

```
17 |     cout<<D(1,3);
   |                ^
```

`#ifdef`, `#endif` – перевірка, чи вже було визначено значення (If Defined). Директива `#endif` є кінцем блоку перевірки `#ifdef`.

`#ifndef` (If Not Defined) – якщо такого значення ще не було.

На прикладі нижче є визначена константа `PRINT_JOE` – на рядку 17 директивою `#ifdef` компілятором перевіряється чи вже було визначено константу `PRINT_JOE`. Вона вже була визначена, отож на рядку 18 виводжу відповідне повідомлення у консоль.

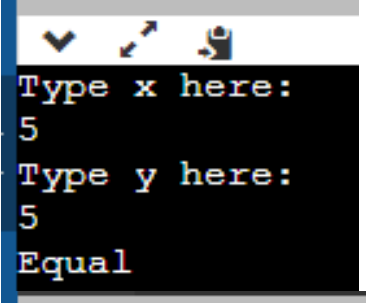
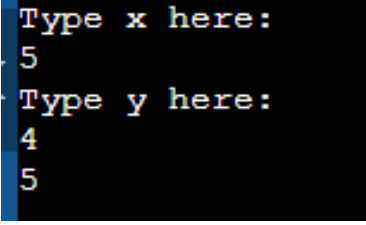


```
13 int main()
14 {
15     #define PRINT_JOE
16
17     #ifdef PRINT_JOE
18     cout<< "Joe" << "\n";
19     #endif
20
21     #ifndef PRINT_BOB
22     cout<< "Not Bob";
23     #endif
24 }
25
```

Joe
Not Bob

Для прикладу використав директиву `#ifndef` – рядок 21 перевіряє чи НЕ було визначено `PRINT_BOB`. Такої константи визначено не було, отже відповідне повідомлення про Боба також виводиться.

3. Скласти програму на мові C++, яка з двох введених з клавіатури цілих чисел відшукує найбільше.

<pre>9 #include <iostream> 10 11 using namespace std; 12 13 int main() 14 { 15 int x, y; 16 cout<<"Type x here: " << endl; 17 cin>>x; 18 cout<<"Type y here: " << endl; 19 cin>>y; 20 21 if(x > y) { 22 cout<<x; 23 } else if (x < y) { 24 cout << y; 25 } else { 26 cout<<"Equal"; 27 } 28 } 29</pre>	 
<pre>Type x here: 2 Type y here: 5 5</pre>	