

Міністерство освіти і науки України
Національний університет «Львівська політехніка»
Інститут телекомунікацій, радіоелектроніки та електронної техніки
кафедра «Радіоелектронні пристрої та системи»

Лабораторна робота №6
з дисципліни «Програмування частина 2»
«Загальна структура програми на мові C»

Мета роботи: ознайомитися із загальною структурою побудови програм на мові C, навчитися використовувати функції введення та виведення даних

Підготував:
ст. групи АП-11
Василюк Ростислав

Прийняв:
Чайковський І.Б.

Львів 2024

Теоретичні відомості

Програма на мові C складається з однієї або більше функцій і хоча б одна з них повинна називатися `main()`. Опис функції складається з заголовку та тіла.

Заголовок у свою чергу містить директиви препроцесора типу `#include` тощо, що під'єднують бібліотечні файли та специфікують перетворення тексту програми перед компіляцією; а також ім'я функції. Ознакою імені функції служать круглі дужки.

Тіло функції поміщається в фігурні дужки та є набором операторів (команд), кожен із яких закінчується символом `“ ; “` - крапка з комою. Елементом програми є коментар - частина тексту програми для пояснення окремих операторів, що входять до її складу.

Коментар не впливає на виконання операторів і записується таким чином :
`//текст коментарю` або так: `/* текст коментарю*/` . В першому випадку коментар має бути єдиним у рядку або в кінці рядка.

Другий спосіб дозволяє записувати коментар будь-де в тексті програми
Оголошення змінної задає ім'я та атрибути змінної. Визначення змінної, крім задання імені та її атрибутів, приводить до виділення для неї пам'яті.

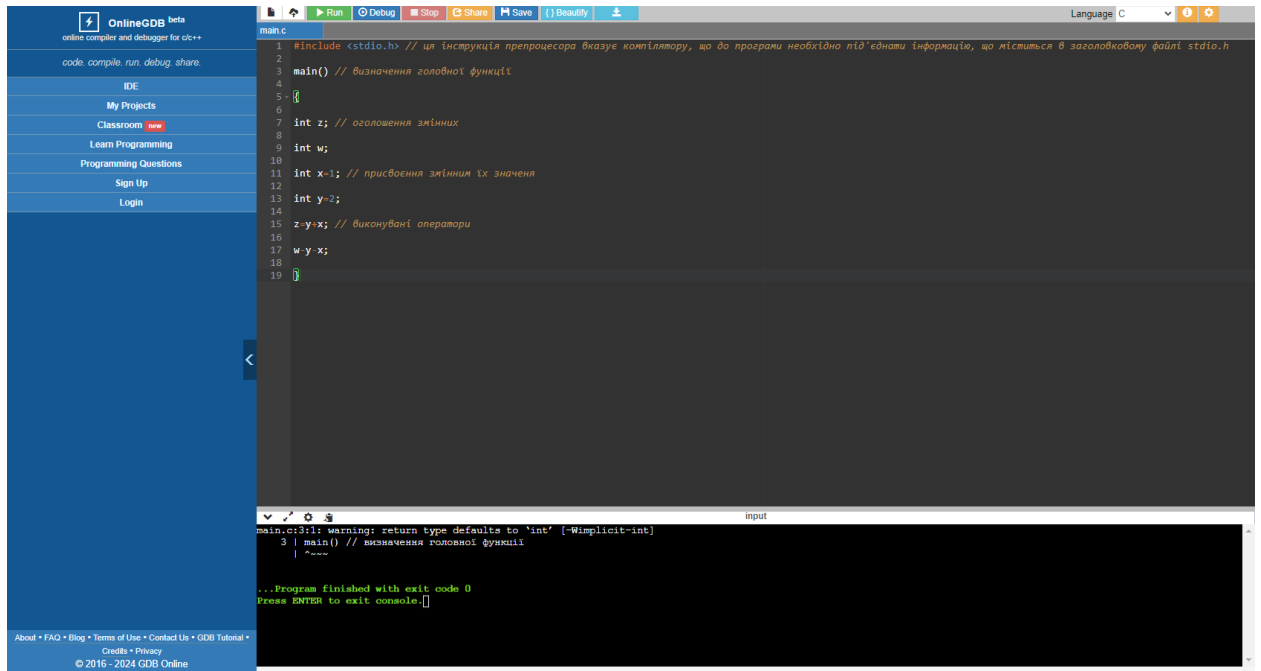
Програма може містити довільне число директив, вказівок компілятору, оголошень та визначень. Їх синтаксис розглядатиметься нижче. Порядок появи цих елементів у програмі є важливий.

Директиви препроцесора – це інструкції, записані в програмі на мові C.

Хід роботи

1. Ознайомитися з теоретичними відомостями.
2. Здійснити виконання усіх прикладів, представлених у теоретичних відомостях, після чого представити їх скріни та результати їх виконання у звіті.

А) Нижче наведена проста програма

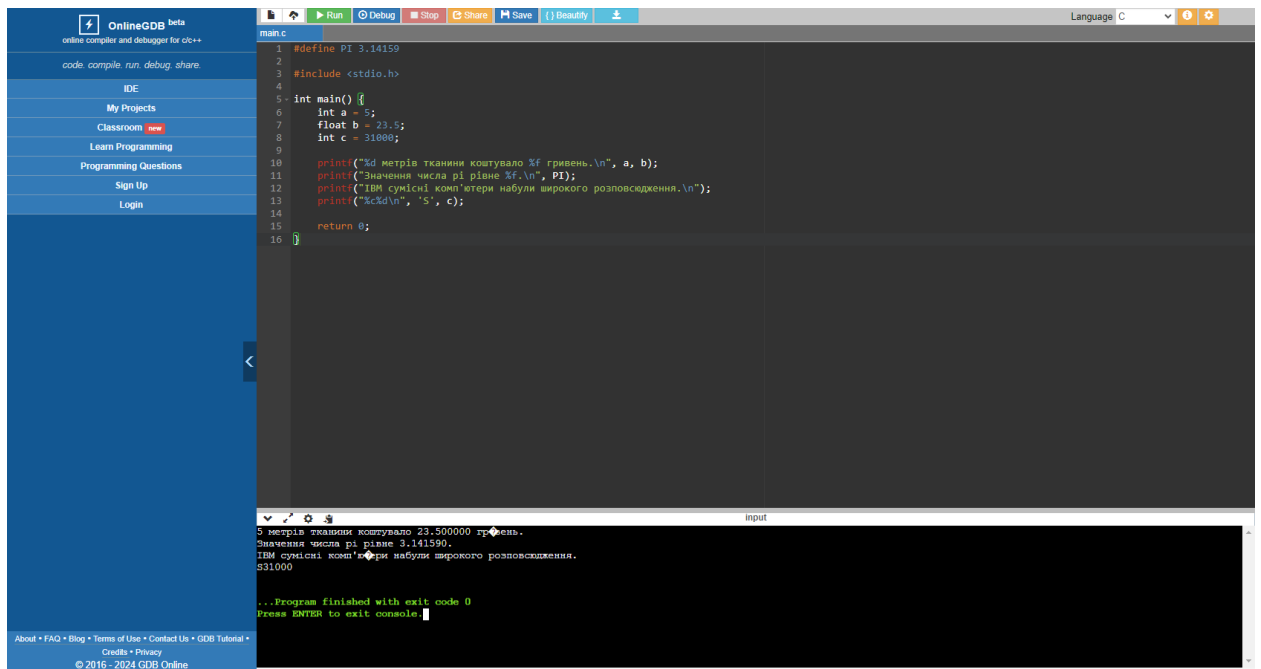


```
main.c
1 #include <stdio.h> // ця інструкція препроцесора вказує компілятору, що до програми необхідно під'єднати інформацію, що міститься в заголовковому файлі stdio.h
2
3 main() // Визначення головної функції
4 {
5     //
6
7     int z; // оголошення змінних
8
9     int w;
10
11     int x=1; // присвоєння змінним їх значення
12
13     int y=2;
14
15     z=y*x; // виконуваний оператор
16
17     w=y*x;
18
19 }
```

main.c:3:11: warning: return type defaults to 'int' [-Wimplicit-int]
3 | main() // визначення головної функції
 | ^~~~~

...Program finished with exit code 0
Press ENTER to exit console.

Б) Використання функції printf ().

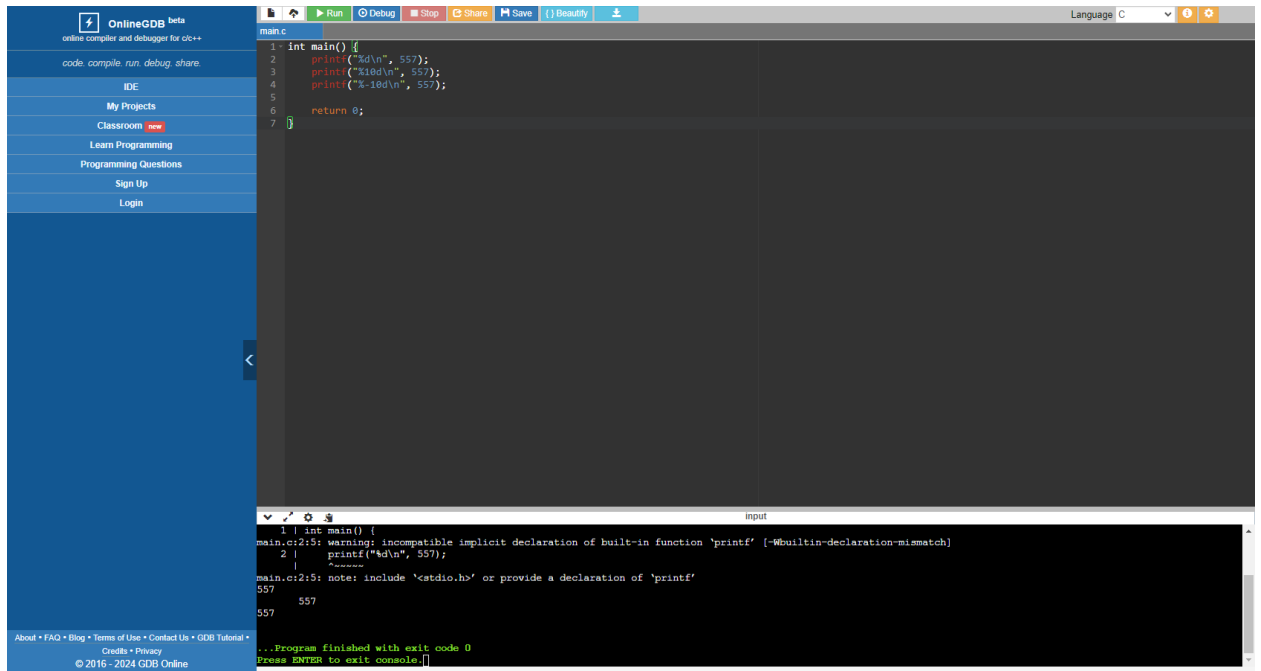


```
main.c
1 #define PI 3.14159
2
3 #include <stdio.h>
4
5 int main() {
6     int a = 5;
7     float b = 23.5;
8     int c = 31000;
9
10     printf("%d метрів тканини коштувало %f гривень.\n", a, b);
11     printf("Значення числа pi рівне %f.\n", PI);
12     printf("IBM сумісні комп'ютери набули широкого розповсюдження.\n");
13     printf("%c%d\n", 'S', c);
14
15     return 0;
16 }
```

5 метрів тканини коштувало 23.500000 гривень.
Значення числа pi рівне 3.141590.
IBM сумісні комп'ютери набули широкого розповсюдження.
S31000

...Program finished with exit code 0
Press ENTER to exit console.

В) Приклади роботи модифікаторів ширини поля на друк цілого числа.



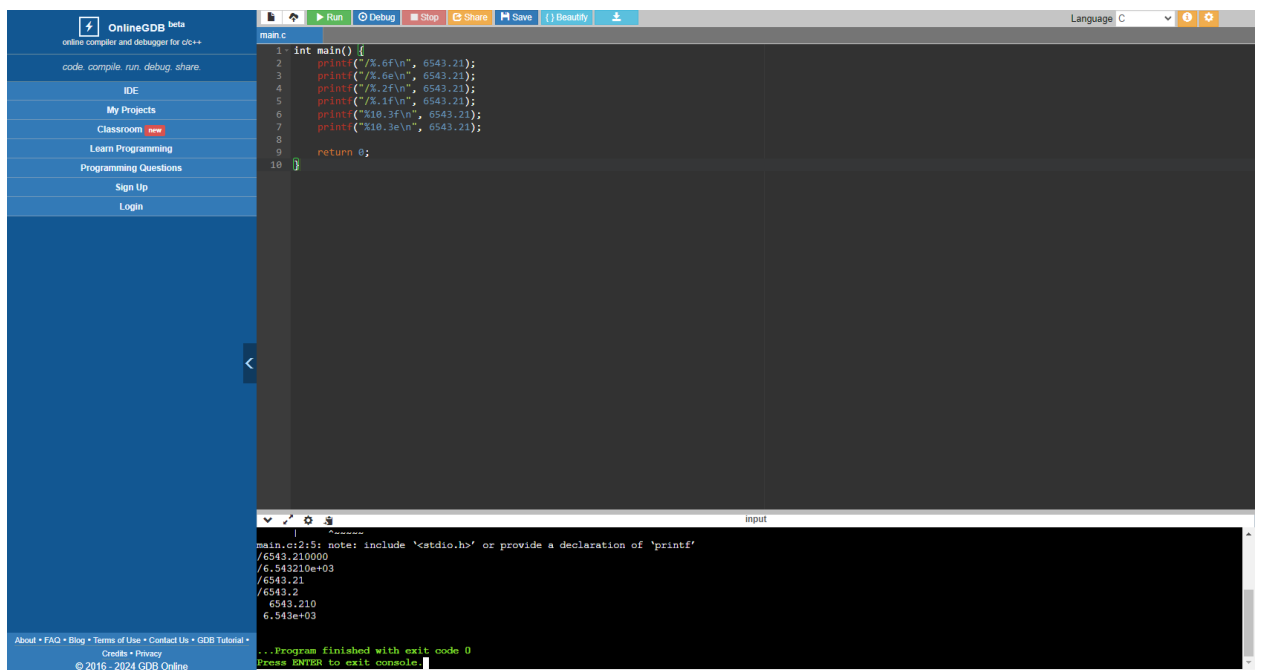
The screenshot shows the OnlineGDB interface with a C program that demonstrates width modifiers for integer printing. The code is as follows:

```
1 int main() {  
2     printf("%d\n", 557);  
3     printf("%10d\n", 557);  
4     printf("%-10d\n", 557);  
5  
6     return 0;  
7 }
```

The output window shows the results of these print statements:

```
1 | int main() {  
main.c:2:5: warning: incompatible implicit declaration of built-in function 'printf' [-Wbuiltin-declaration-mismatch]  
2 |     printf("%d\n", 557);  
   |     ~~~~~  
main.c:2:5: note: include '<stdio.h>' or provide a declaration of 'printf'  
557  
557  
...Program finished with exit code 0  
Press ENTER to exit console.[]
```

Г) Розглянемо приклади форматів, що відповідають даним з плаваючою точкою.



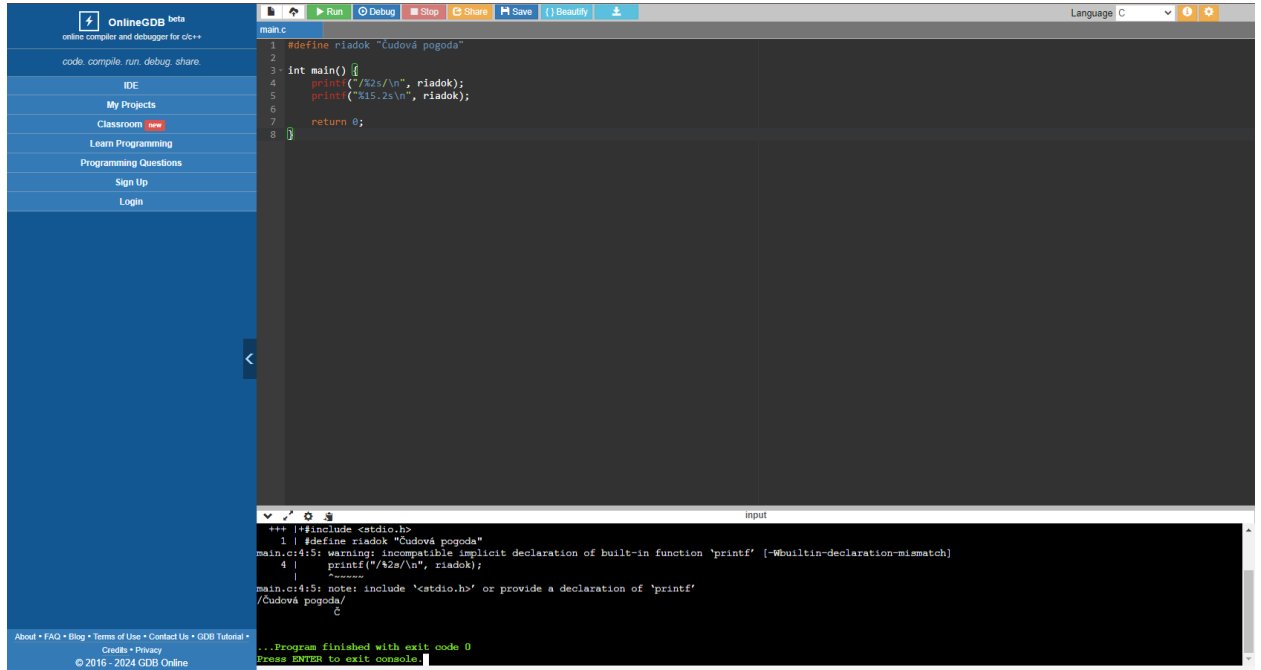
The screenshot shows the OnlineGDB interface with a C program that demonstrates various floating-point format specifiers. The code is as follows:

```
1 int main() {  
2     printf("%f\n", 6543.21);  
3     printf("%e\n", 6543.21);  
4     printf("%2f\n", 6543.21);  
5     printf("%1f\n", 6543.21);  
6     printf("%10.3f\n", 6543.21);  
7     printf("%10.3e\n", 6543.21);  
8  
9     return 0;  
10 }
```

The output window shows the results of these print statements:

```
main.c:2:5: note: include '<stdio.h>' or provide a declaration of 'printf'  
/6543.210000  
/6.543210e+03  
/6543.21  
/6543.2  
6543.210  
6.543e+03  
...Program finished with exit code 0  
Press ENTER to exit console.[]
```

Д) Розглянемо застосування функції printf() для роботи із рядками.



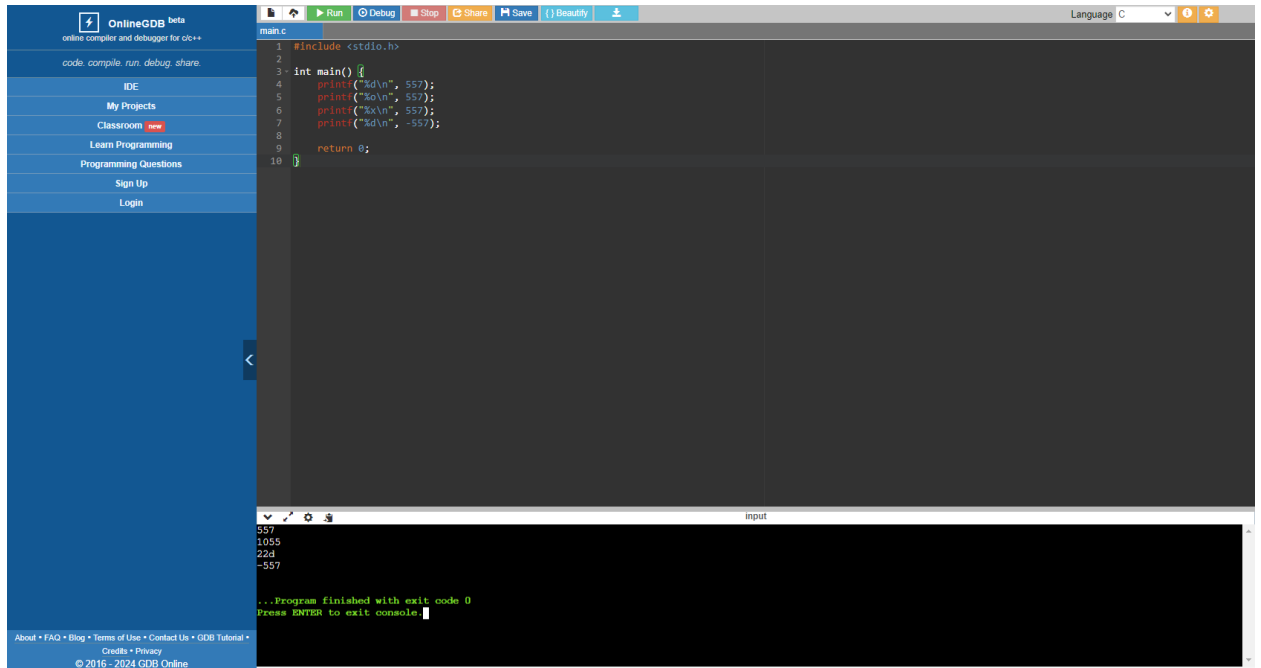
The screenshot shows the OnlineGDB interface with a C program in the editor. The program defines a string 'riadok' and prints it using two different printf format specifiers. The console output shows the string 'čudová pogoda' printed twice, once with a default width and once with a width of 15 and 2 decimal places. The console also displays a warning about an incompatible implicit declaration of 'printf' and a note to include 'stdio.h'.

```
main.c
1 | #define riadok "čudová pogoda"
2 |
3 | int main() {
4 |     printf("/%2s/\n", riadok);
5 |     printf("/%15.2s\n", riadok);
6 |
7 |     return 0;
8 | }
```

```
+++ | #include <stdio.h>
1 | #define riadok "čudová pogoda"
main.c:4:5: warning: incompatible implicit declaration of built-in function 'printf' [-Wbuiltin-declaration-mismatch]
4 |     printf("/%2s/\n", riadok);
  |     ~~~~~
main.c:4:5: note: include 'stdio.h' or provide a declaration of 'printf'
/čudová pogoda/
č

...Program finished with exit code 0
Press ENTER to exit console
```

Е) Розглянемо застосування функції printf() для перетворення дани



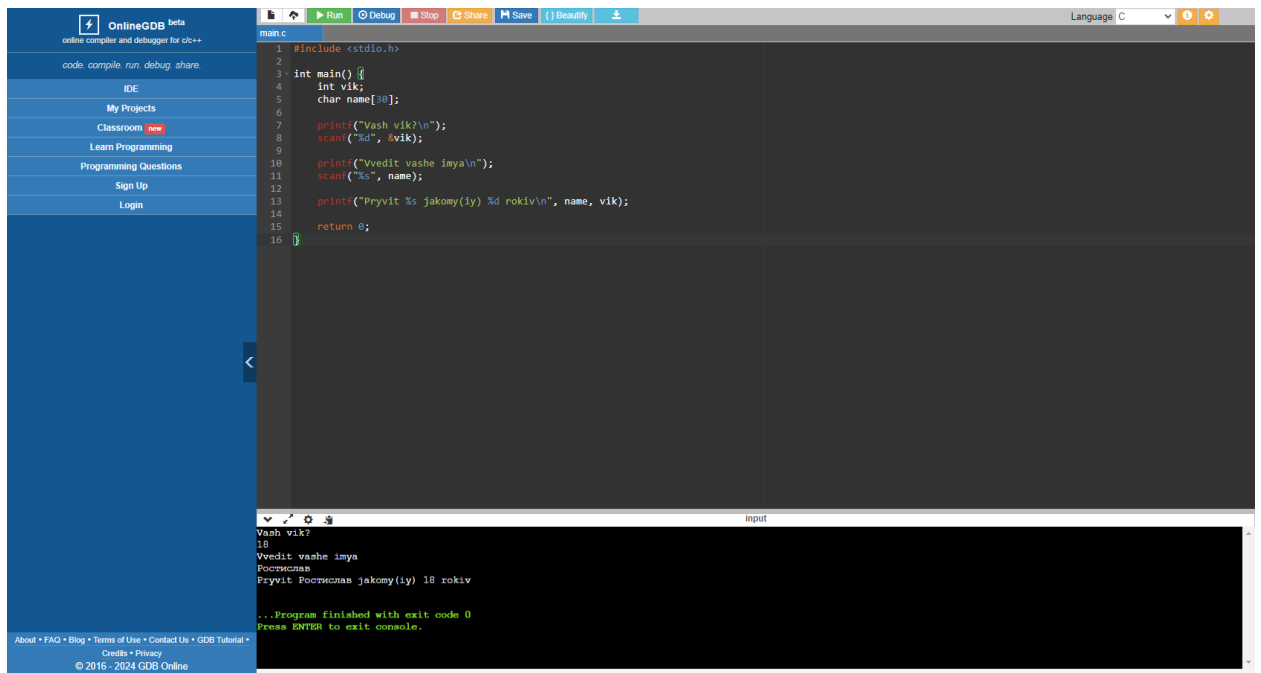
The screenshot shows the OnlineGDB interface with a C program in the editor. The program includes 'stdio.h' and prints the integer value 557 using five different printf format specifiers: %d, %i, %o, %x, and %u. The console output shows the value 557 printed five times in different formats: decimal, integer, octal, hexadecimal, and unsigned decimal. The console also displays the program finished with exit code 0.

```
main.c
1 | #include <stdio.h>
2 |
3 | int main() {
4 |     printf("%d\n", 557);
5 |     printf("%i\n", 557);
6 |     printf("%o\n", 557);
7 |     printf("%x\n", 557);
8 |     printf("%u\n", 557);
9 |     return 0;
10 | }
```

```
557
1055
223
~557

...Program finished with exit code 0
Press ENTER to exit console
```

Є) Розглянемо програму, що демонструє використання функцій `printf()` і `scanf()`.



The screenshot shows the OnlineGDB interface with a C program that uses `printf()` and `scanf()`. The code is as follows:

```
1 #include <stdio.h>
2
3 int main() {
4     int vik;
5     char name[30];
6
7     printf("Vash vik?\n");
8     scanf("%d", &vik);
9
10    printf("Vvedit vashe imya\n");
11    scanf("%s", name);
12
13    printf("Privit %s jakomy(iy) %d rokiv\n", name, vik);
14
15    return 0;
16 }
```

The console output shows the program's execution:

```
Vash vik?
18
Vvedit vashe imya
Ростислав

Privit Ростислав jakomy(iy) 18 rokiv

...Program finished with exit code 0
Press ENTER to exit console.
```

Ж) Наступна програма демонструє виведення на друк групи символів із припиненням роботи програми при введенні певного символу



The screenshot shows the OnlineGDB interface with a C program that uses `getchar()` and `putchar()`. The code is as follows:

```
1 #include <stdio.h>
2 #include <conio.h>
3
4 #define STOP '!'
5
6 int main() {
7     char ch;
8
9
10    ch = getchar();
11
12
13    m1:
14
15    if (ch != STOP) {
16        putchar(ch);
17        ch = getchar();
18        goto m1;
19    }
20
21    return 0;
22 }
23
24
25 }
```

The console output shows the program's execution:

```
aaa
aaa
123
Привет
!
```

3. Оформити звіт.

Контрольні питання

1. Структура програми на мові C.

Програма на мові C складається з наступних основних частин: Директиви препроцесора (`include`, `define` тощо)

Функція `main()`, яка є точкою входу в програму

Інші функції, які викликаються з `main()`

Змінні, глобальні та локальні

2. Ідеологія організації операцій введення-виведення в мові C.

Мова C надає низькорівневий доступ до операцій введення-виведення, використовуючи файлові потоки. Основні функції: `scanf()`, `printf()`, `getchar()`, `putchar()`.

3. Синтаксис функцій `printf()` і `scanf()`.

`printf("format_string", args);`

`scanf("format_string", &vars);`

4. Основні типи форматів при звертанні до функцій `printf()` і `scanf()`.

`%d` - ціле число

`%f` - число з плаваючою комою

`%c` - символ

`%s` - рядок

`%x`, `%X` - шістнадцяткове число

5. Модифікатори форматів при звертанні до функцій `printf()` і `scanf()`.

`%5d` - ціле число, вирівняне по правому краю, 5 символів

`%.2f` - число з плаваючою комою, 2 знаки після коми

`%10s` - рядок, вирівняний по правому краю, 10 символів

6. Відмінності при застосуванні функцій `printf()` і `scanf()`.

`printf()` - виводить дані на екран

`scanf()` - зчитує дані з клавіатури, записуючи їх у вказані змінні

7. Застосування функцій `getchar()` і `putchar()`.

`getchar()` - зчитує один символ з клавіатури

`putchar()` - виводить один символ на екран

8. Пояснити зміст і обґрунтувати результати виконаних прикладів.

Програми демонструють використання основних функцій вводу-виводу в мові C: `printf()`, `scanf()`, `getchar()`, `putchar()`. Показано, як використовувати різні формати та модифікатори при виведенні даних за допомогою `printf()`, а також зчитування даних з клавіатури за допомогою `scanf()`. Також наведено приклад використання `getchar()` і `putchar()` для зчитування та виведення окремих символів.