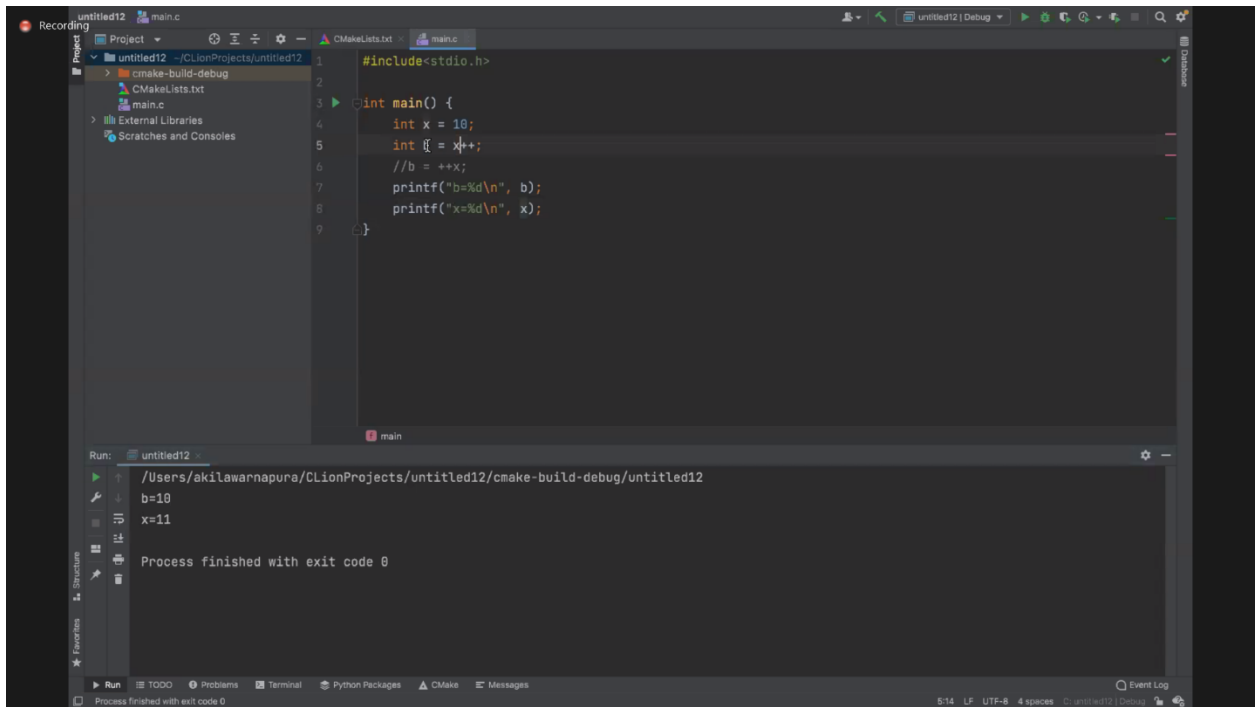


Exercise.

01.

a)



```
#include <stdio.h>

int main() {
    int x = 10;
    int b = x++;
    // b = ++x;
    printf("b=%d\n", b);
    printf("x=%d\n", x);
}
```

Run: untitled12

/Users/akilawannapura/CLionProjects/untitled12/cmake-build-debug/untitled12

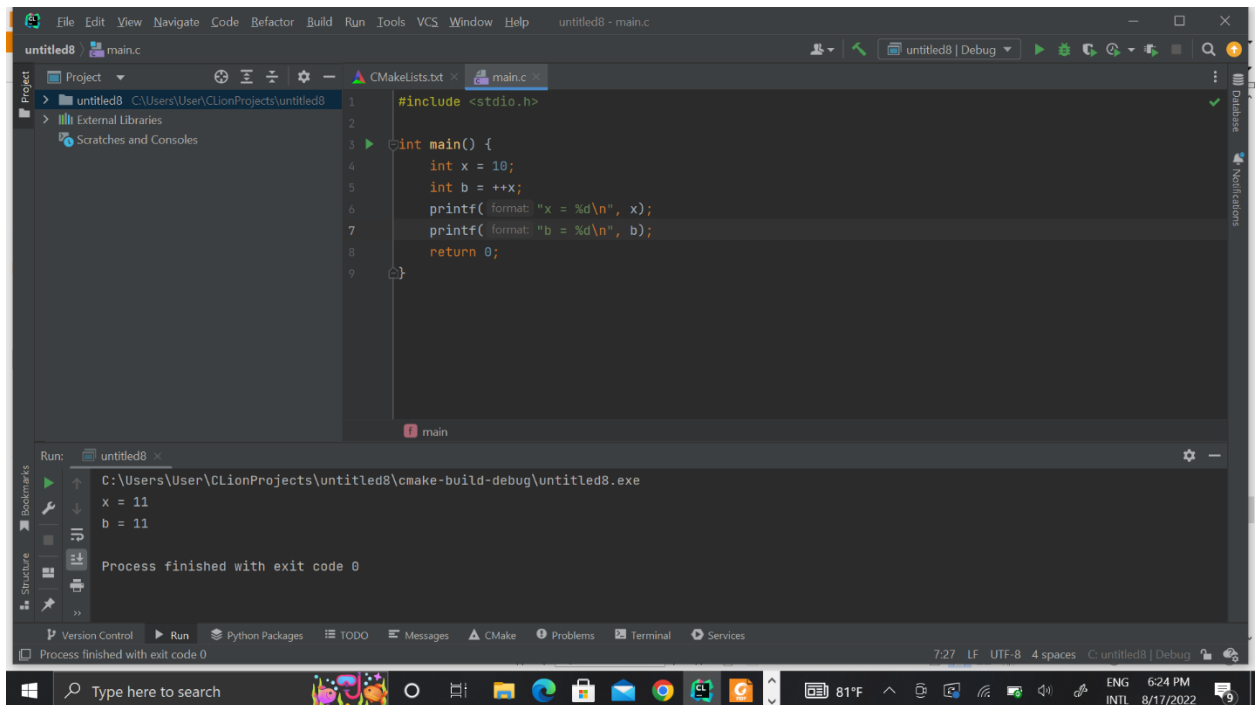
b=10
x=11

Process finished with exit code 0

→ ∴ int x = 10;
∴ int b = x++; → ∴ x = 11
∴ printf("x = %d\n", x); ∴ b = 10
∴ printf("b = %d\n", b);

∴ This code, first x value assigned to b and after that x value increase in postfix operator.

b)



```
#include <stdio.h>

int main() {
    int x = 10;
    int b = ++x;
    printf("x = %d\n", x);
    printf("b = %d\n", b);
    return 0;
}
```

Run: untitled8.exe
C:\Users\User\CLionProjects\untitled8\cmake-build-debug\untitled8.exe
x = 11
b = 11
Process finished with exit code 0

→ ∴ int x = 10 ;
∴ int b = ++x ; ⇒ ∴ x = 11
∴ printf (" x = %d \n " , x) ; ∴ b = 11
∴ printf (" b = %d \n " , b) ;

∴ This code, first increase the x value and after that, after x value assigned to b in prefix operator.

02.

The screenshot shows the Visual Studio Code interface with a project named 'untitled8'. The main editor displays the following C code in 'main.c':

```
1 #include <stdio.h>
2
3 int main() {
4     int x = 5 + 3 * 8 / 2 + 2;
5     printf("format: 'x = %d\n", x);
6     return 0;
7 }
```

The Run and Debug console at the bottom shows the execution output:

```
Run: untitled8
C:\Users\User\CLionProjects\untitled8\cmake-build-debug\untitled8.exe
x = 19
Process finished with exit code 0
```

The status bar at the bottom indicates the build finished in 672 ms (2 minutes ago).

03.

The screenshot shows the Visual Studio Code interface with a project named 'untitled8'. The main editor displays the following C code in 'main.c':

```
1 #include <stdio.h>
2
3 int main() {
4     int i = 5;
5     printf("format: '%d', (++i)++);
6     return 0;
7 }
```

The Messages panel at the bottom shows the build output:

```
Build
===== [ Build | untitled8 | Debug ] =====
"C:\Program Files\JetBrains\CLion 2022.1.3\bin\cmake\win\bin\cmake.exe" --build C:\Users\User\CLionProjects\untitled8\cmake-build-debug --target unt
[1/2] Building C object CMakeFiles/untitled8.dir/main.c.obj
FAILED: CMakeFiles/untitled8.dir/main.c.obj
C:\PROGRA~1\JETBRA~1\CLION2~1.3\bin\mingw\bin\gcc.exe -g -std=gnu99 -MD -MT CMakeFiles/untitled8.dir/main.c.obj -MF CMakeFiles\untitled8.dir\main.
C:\Users\User\CLionProjects\untitled8\main.c: In function 'main':
C:\Users\User\CLionProjects\untitled8\main.c:5:23: error: lvalue required as increment operand
5 |     printf("format: '%d', (++i)++);
  |                       ^~
ninja: build stopped: subcommand failed.
```

The status bar at the bottom indicates the build failed in 396 ms (a minute ago).

04.

The screenshot shows a CMake IDE with a project named 'untitled8'. The main.c file contains the following code:

```
#include <stdio.h>

int main() {
    int i = 12;
    int j = sizeof(i++);
    printf("format: \"%d %d\", i, j);
    return 0;
}
```

The Run window shows the execution of the program, outputting '12 4' and 'Process finished with exit code 0'.

No: _____ Date: ____/____/____

→ ∴ int i = 12 ;
 ∴ int j = i++ ;

⊕
 ↓

0	-----	000000001100
		+ 1
0	-----	000000001101

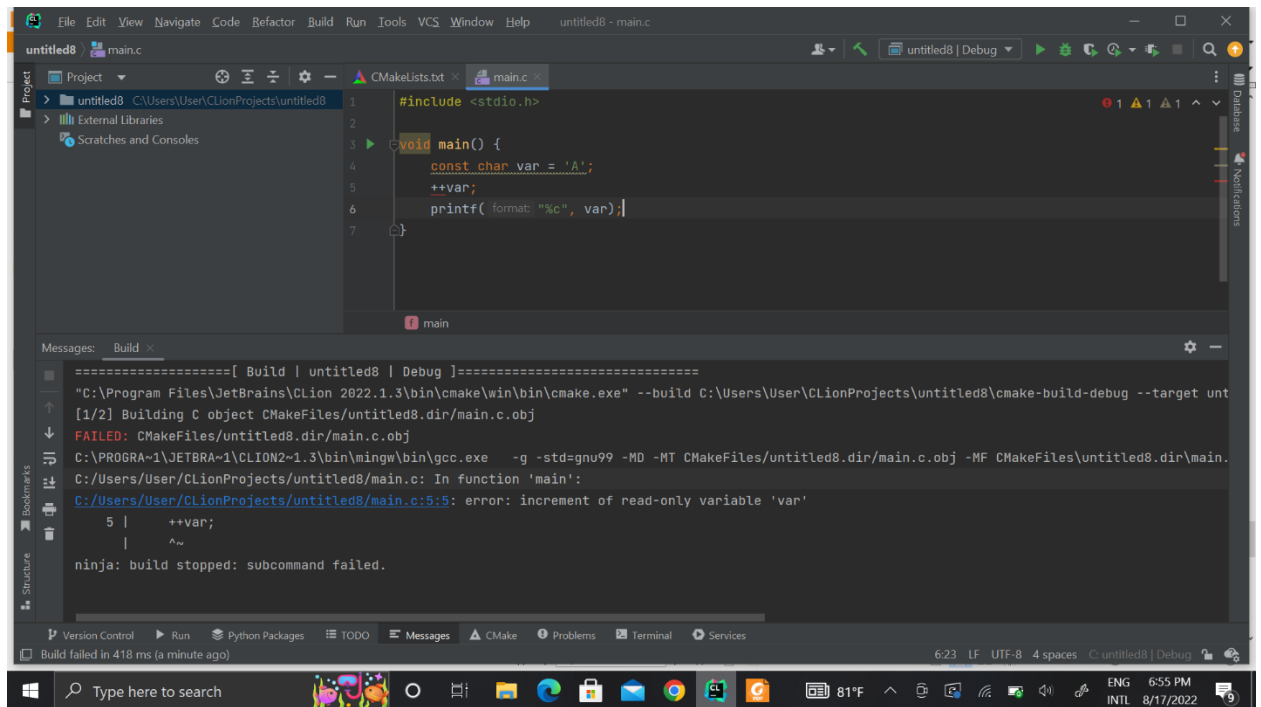
2³ 2² 2¹ 2⁰

→ 2³ × 1 + 2² × 1 +
 2¹ × 0 + 2⁰ × 1
 = 13

∴ Integer value has 4 bytes.
 ∴ int j = sizeof(i++) ;
 _____ → 4

05.

a)



The screenshot shows the CLion IDE interface. The main editor displays a C program in `main.c`:

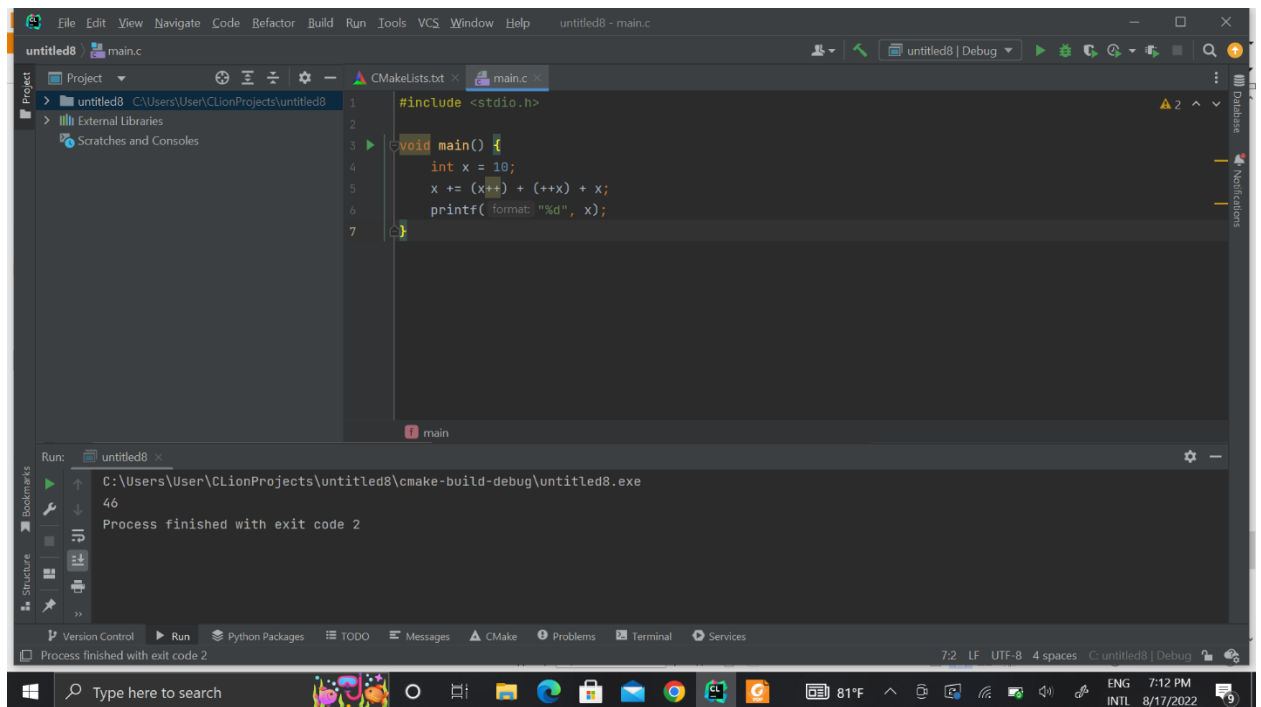
```
1 #include <stdio.h>
2
3 void main() {
4     const char var = 'A';
5     ++var;
6     printf( format: "%c", var);
7 }
```

The `main` function is highlighted in the `Structure` tool window. The `Messages` panel at the bottom shows the following error:

```
===== [ Build | untitled8 | Debug ] =====
"C:\Program Files\JetBrains\CLion 2022.1.3\bin\cmake\win\bin\cmake.exe" --build C:\Users\User\CLionProjects\untitled8\cmake-build-debug --target unt
[1/2] Building C object CMakeFiles/untitled8.dir/main.c.obj
FAILED: CMakeFiles/untitled8.dir/main.c.obj
C:\PROGRA~1\JETBRA~1\CLION2~1.3\bin\mingw\bin\gcc.exe -g -std=gnu99 -MD -MT CMakeFiles/untitled8.dir/main.c.obj -MF CMakeFiles\untitled8.dir/main.
C:/Users/User/CLionProjects/untitled8/main.c:5:5: error: increment of read-only variable 'var'
5 |     ++var;
  |     ^~
ninja: build stopped: subcommand failed.
```

The status bar at the bottom indicates the build failed in 418 ms.

b)



The screenshot shows the CLion IDE interface after the program has been executed. The main editor displays the same C program as in part (a):

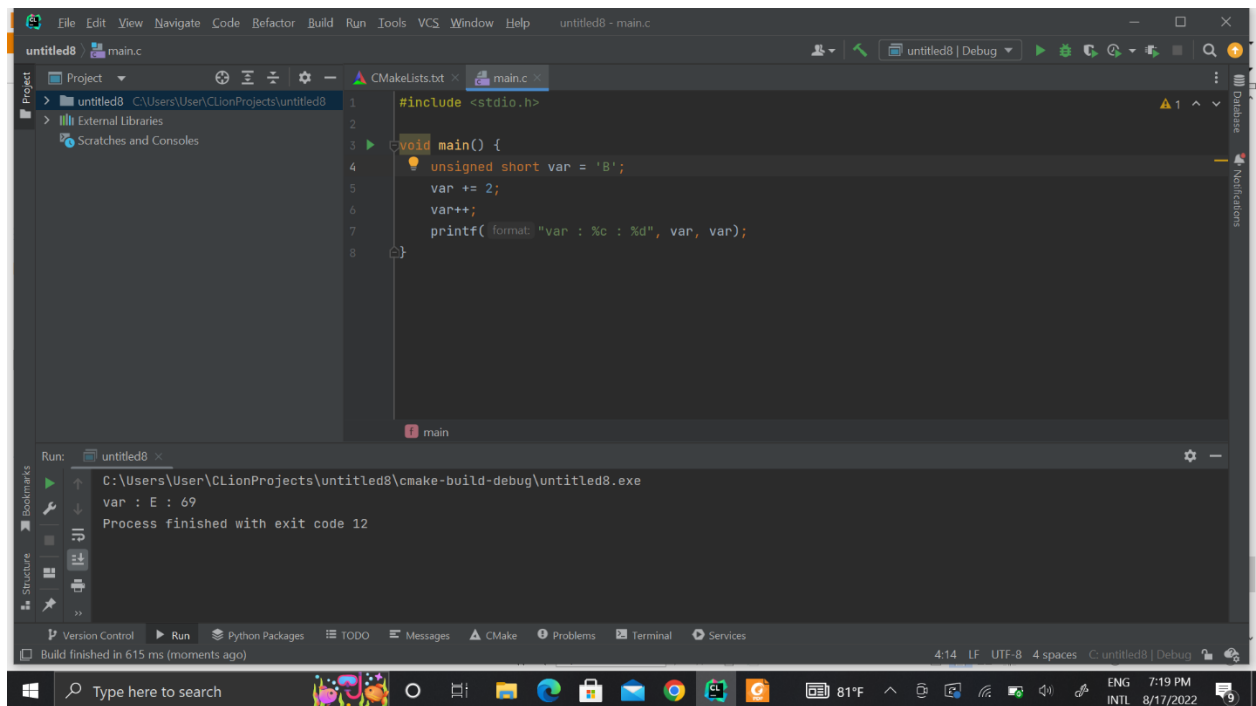
```
1 #include <stdio.h>
2
3 void main() {
4     int x = 10;
5     x += (x++) + (++x) + x;
6     printf( format: "%d", x);
7 }
```

The `main` function is highlighted in the `Structure` tool window. The `Run` panel at the bottom shows the execution details:

```
Run: untitled8 x
C:\Users\User\CLionProjects\untitled8\cmake-build-debug\untitled8.exe
46
Process finished with exit code 2
```

The status bar at the bottom indicates the process finished with exit code 2.

c)



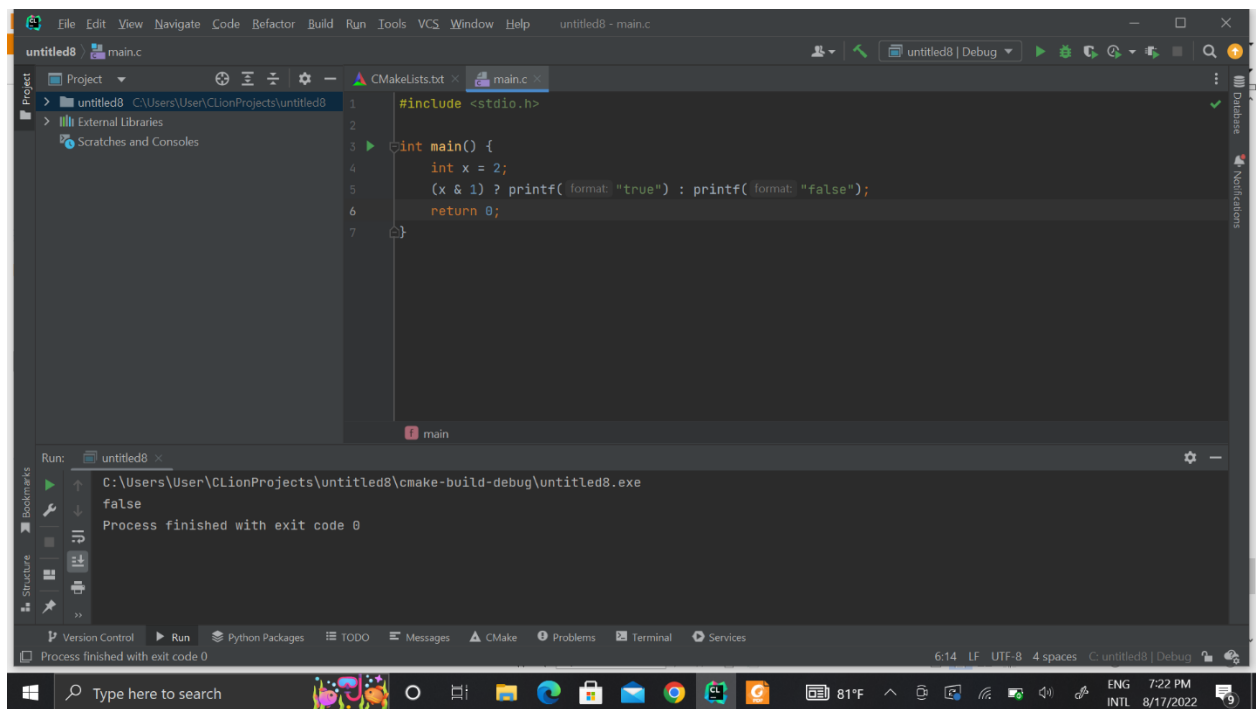
The screenshot shows the Visual Studio Code interface with a C program in `main.c`. The code includes `<stdio.h>` and defines a `main` function. Inside `main`, an `unsigned short` variable `var` is initialized to `'B'`, incremented by 2, and then printed using `printf`. The Run and Debug console shows the output `var : E : 69` and a process finished with exit code 12.

```
#include <stdio.h>

void main() {
    unsigned short var = 'B';
    var += 2;
    var++;
    printf( format: "var : %c : %d", var, var);
}
```

Run: untitled8
C:\Users\User\CLionProjects\untitled8\cmake-build-debug\untitled8.exe
var : E : 69
Process finished with exit code 12

d)



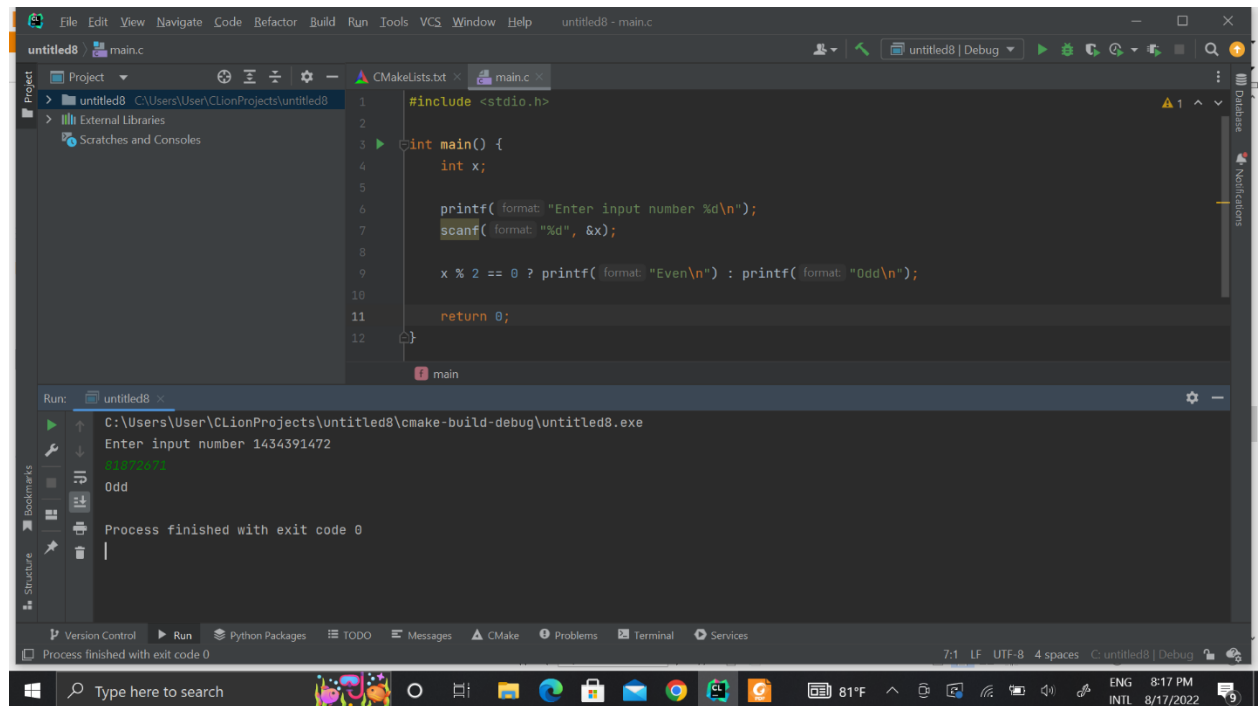
The screenshot shows the Visual Studio Code interface with a C program in `main.c`. The code includes `<stdio.h>` and defines a `main` function. Inside `main`, an `int` variable `x` is initialized to 2. A conditional statement checks if `x` is even (`x % 2 == 0`) and prints "true" or "false". The program then returns 0. The Run and Debug console shows the output `false` and a process finished with exit code 0.

```
#include <stdio.h>

int main() {
    int x = 2;
    (x % 2 == 0) ? printf( format: "true") : printf( format: "false");
    return 0;
}
```

Run: untitled8
C:\Users\User\CLionProjects\untitled8\cmake-build-debug\untitled8.exe
false
Process finished with exit code 0

06)



```
#include <stdio.h>

int main() {
    int x;

    printf( format: "Enter input number %d\n");
    scanf( format: "%d", &x);

    x % 2 == 0 ? printf( format: "Even\n") : printf( format: "Odd\n");

    return 0;
}
```

Run: untitled8 x

C:\Users\User\CLionProjects\untitled8\cmake-build-debug\untitled8.exe

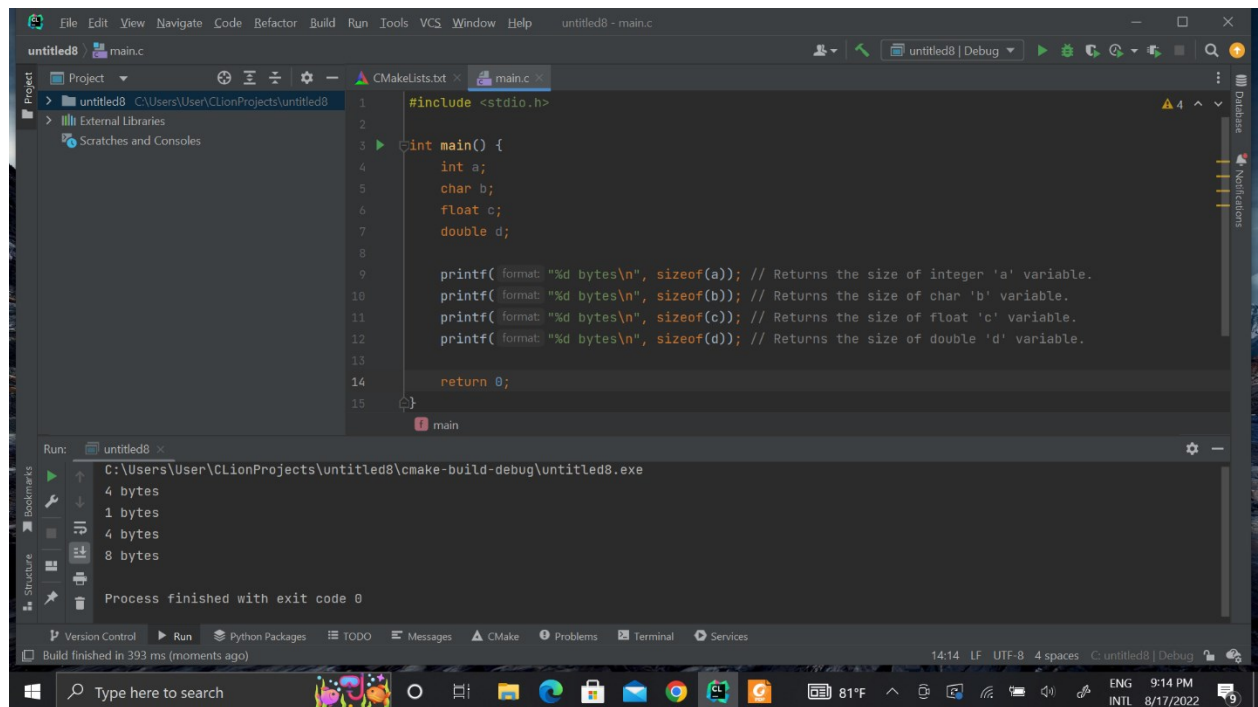
Enter input number 1434391472

81872672

Odd

Process finished with exit code 0

07)



```
#include <stdio.h>

int main() {
    int a;
    char b;
    float c;
    double d;

    printf( format: "%d bytes\n", sizeof(a)); // Returns the size of integer 'a' variable.
    printf( format: "%d bytes\n", sizeof(b)); // Returns the size of char 'b' variable.
    printf( format: "%d bytes\n", sizeof(c)); // Returns the size of float 'c' variable.
    printf( format: "%d bytes\n", sizeof(d)); // Returns the size of double 'd' variable.

    return 0;
}
```

Run: untitled8 x

C:\Users\User\CLionProjects\untitled8\cmake-build-debug\untitled8.exe

4 bytes

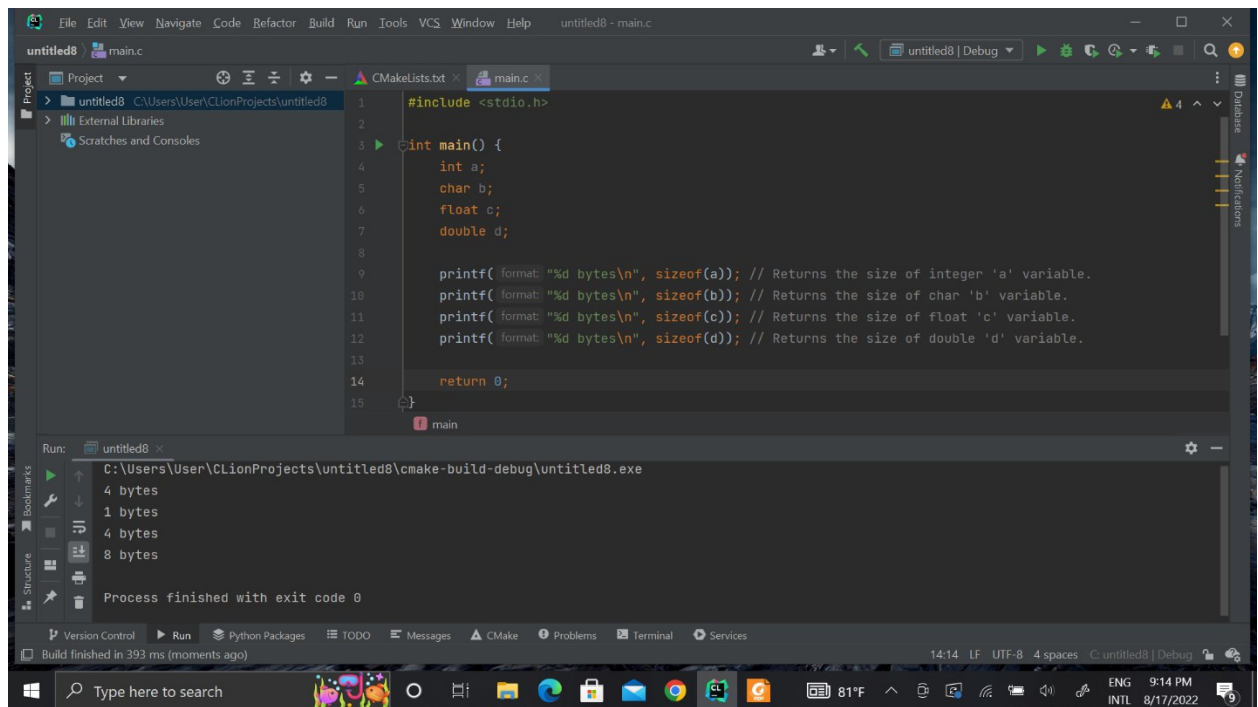
1 bytes

4 bytes

8 bytes

Process finished with exit code 0

08)



The screenshot shows the Visual Studio Code editor with a C program named `main.c`. The program includes `<stdio.h>` and defines a `main` function. Inside `main`, four variables are declared: `int a;`, `char b;`, `float c;`, and `double d;`. Four `printf` statements are used to print the size of each variable in bytes, with comments explaining that `sizeof` returns the size of the variable. The output window shows the results: 4 bytes for `int`, 1 byte for `char`, 4 bytes for `float`, and 8 bytes for `double`. The process finished with exit code 0.

```
#include <stdio.h>

int main() {
    int a;
    char b;
    float c;
    double d;

    printf( format: "%d bytes\n", sizeof(a)); // Returns the size of integer 'a' variable.
    printf( format: "%d bytes\n", sizeof(b)); // Returns the size of char 'b' variable.
    printf( format: "%d bytes\n", sizeof(c)); // Returns the size of float 'c' variable.
    printf( format: "%d bytes\n", sizeof(d)); // Returns the size of double 'd' variable.

    return 0;
}
```

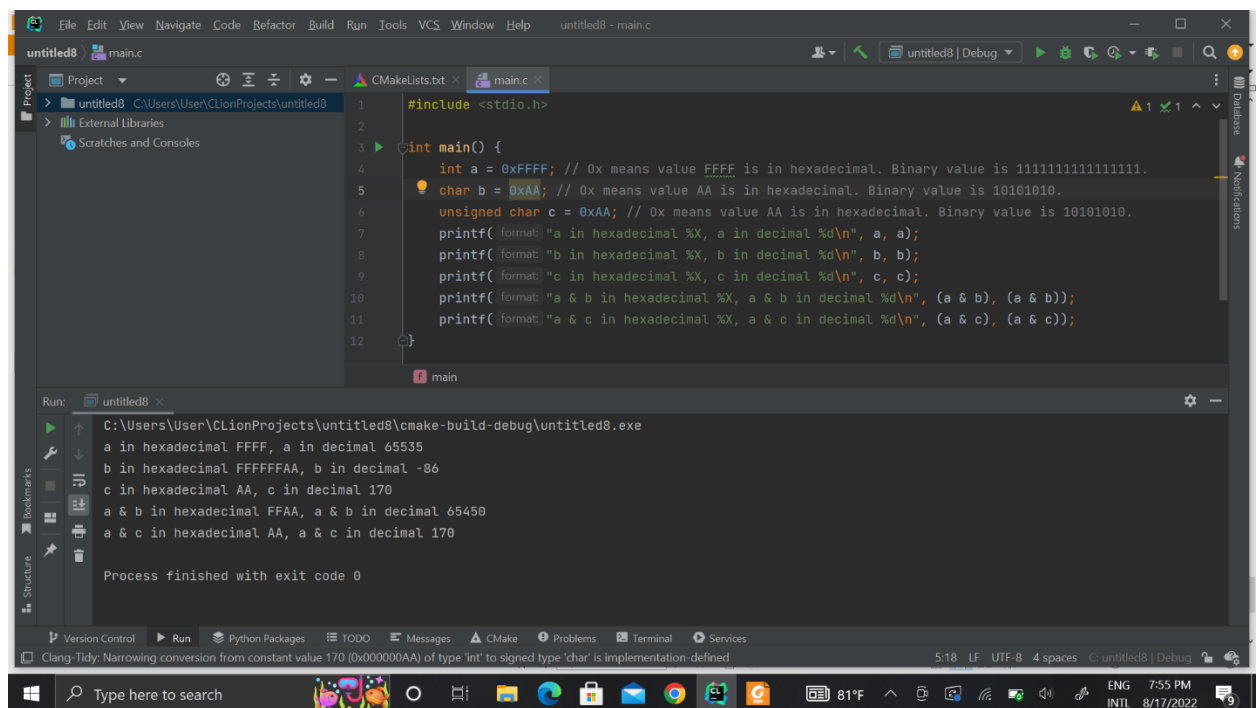
Run: untitled8

C:\Users\User\CLionProjects\untitled8\cmake-build-debug\untitled8.exe

4 bytes
1 bytes
4 bytes
8 bytes

Process finished with exit code 0

09)



The screenshot shows the Visual Studio Code editor with a C program named `main.c`. The program includes `<stdio.h>` and defines a `main` function. Inside `main`, three variables are declared and initialized: `int a = 0xFFFF;` (commented as 0x means value FFFF is in hexadecimal, binary value is 1111111111111111), `char b = 0xAA;` (commented as 0x means value AA is in hexadecimal, binary value is 10101010), and `unsigned char c = 0xAA;` (commented as 0x means value AA is in hexadecimal, binary value is 10101010). Five `printf` statements are used to print the values of `a`, `b`, and `c` in both hexadecimal and decimal formats, as well as their bitwise AND results. The output window shows the results: `a` is 0xFFFF (65535), `b` is 0xFFFFAA (-86), `c` is 0xAA (170), `a & b` is 0xFFAA (65498), and `a & c` is 0xAA (170). The process finished with exit code 0.

```
#include <stdio.h>

int main() {
    int a = 0xFFFF; // 0x means value FFFF is in hexadecimal. Binary value is 1111111111111111.
    char b = 0xAA; // 0x means value AA is in hexadecimal. Binary value is 10101010.
    unsigned char c = 0xAA; // 0x means value AA is in hexadecimal. Binary value is 10101010.

    printf( format: "a in hexadecimal %X, a in decimal %d\n", a, a);
    printf( format: "b in hexadecimal %X, b in decimal %d\n", b, b);
    printf( format: "c in hexadecimal %X, c in decimal %d\n", c, c);
    printf( format: "a & b in hexadecimal %X, a & b in decimal %d\n", (a & b), (a & b));
    printf( format: "a & c in hexadecimal %X, a & c in decimal %d\n", (a & c), (a & c));
}
```

Run: untitled8

C:\Users\User\CLionProjects\untitled8\cmake-build-debug\untitled8.exe

a in hexadecimal FFFF, a in decimal 65535
b in hexadecimal FFFFFFFA, b in decimal -86
c in hexadecimal AA, c in decimal 170
a & b in hexadecimal FFAA, a & b in decimal 65498
a & c in hexadecimal AA, a & c in decimal 170

Process finished with exit code 0