Домашнє завдання №24_1

Застосовуючи OpenCV скласти програму (C++), яка дозволяє виявляти на фото обличчя людей. При цьому потрібно виявляти не більше MAX_FACES_COUNT обличь.

**** коментар:** далі наводиться приклад повністю виконаного завдання; для компіляції на ПК слід скачати з https://opencv.org/releases/ або з https://github.com/opencv/opencv/releases та інсталювати бібліотеку OpenCV(версія 3.4.8), після чого налаштувати Visual Studio як показано в цій відео-демонстрації https://www.youtube.com/watch?v=BqcL-VH7Pag (додатково потрібно буде скопіювати dll-файли).*

Вибір варіанту

(Nж + Nг + 1) % 10 + 1

де: Nж – порядковий номер студента в групі, а Nг – номер групи(1,2,3,4,5,6,7,8 або 9)

Варіанти завдань

Номер **варіанту** відповідає максимально допустимій кількості облич(MAX_FACES_COUNT), які можуть бути виявлена на фото.

Спосіб виконання наведеного прикладу коду за допомогою Visual Studio

На рисунках 1, 2, 3, 5, 6, 7, 8, 9 та 10 послідовно показано налаштування проекту у Visual Studio після інсталяції OpenCV **версії 3.4.8**. На рисунках 11, 12, 13 та 14 показано виконання програми.

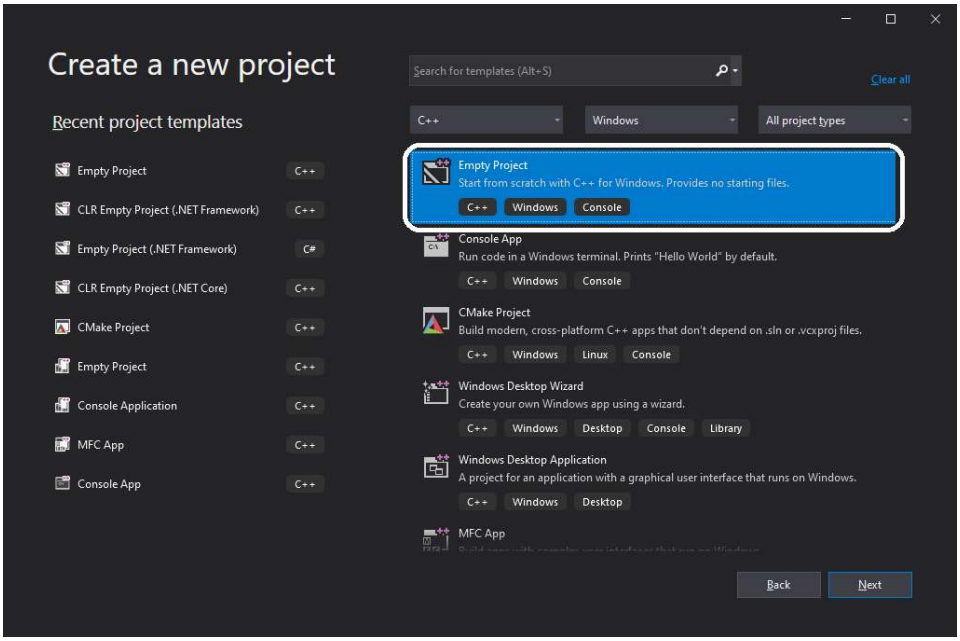

Рис. 1. Створення нового проекту у Visual Studio

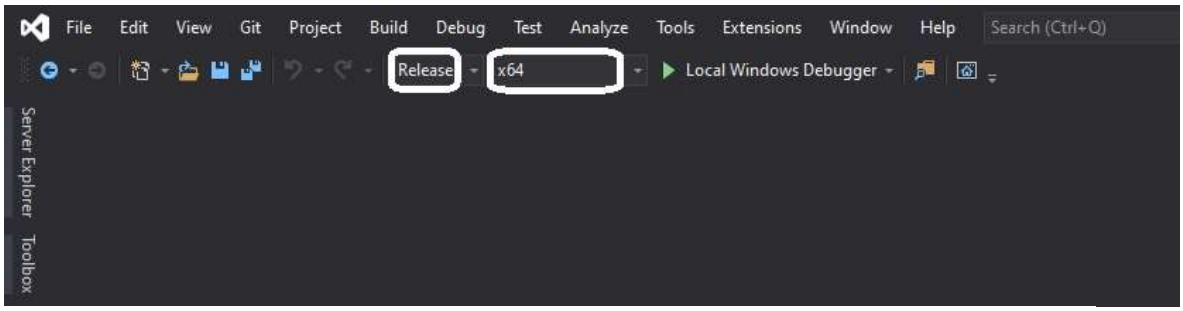

Рис. 2.Вибір конфігурації

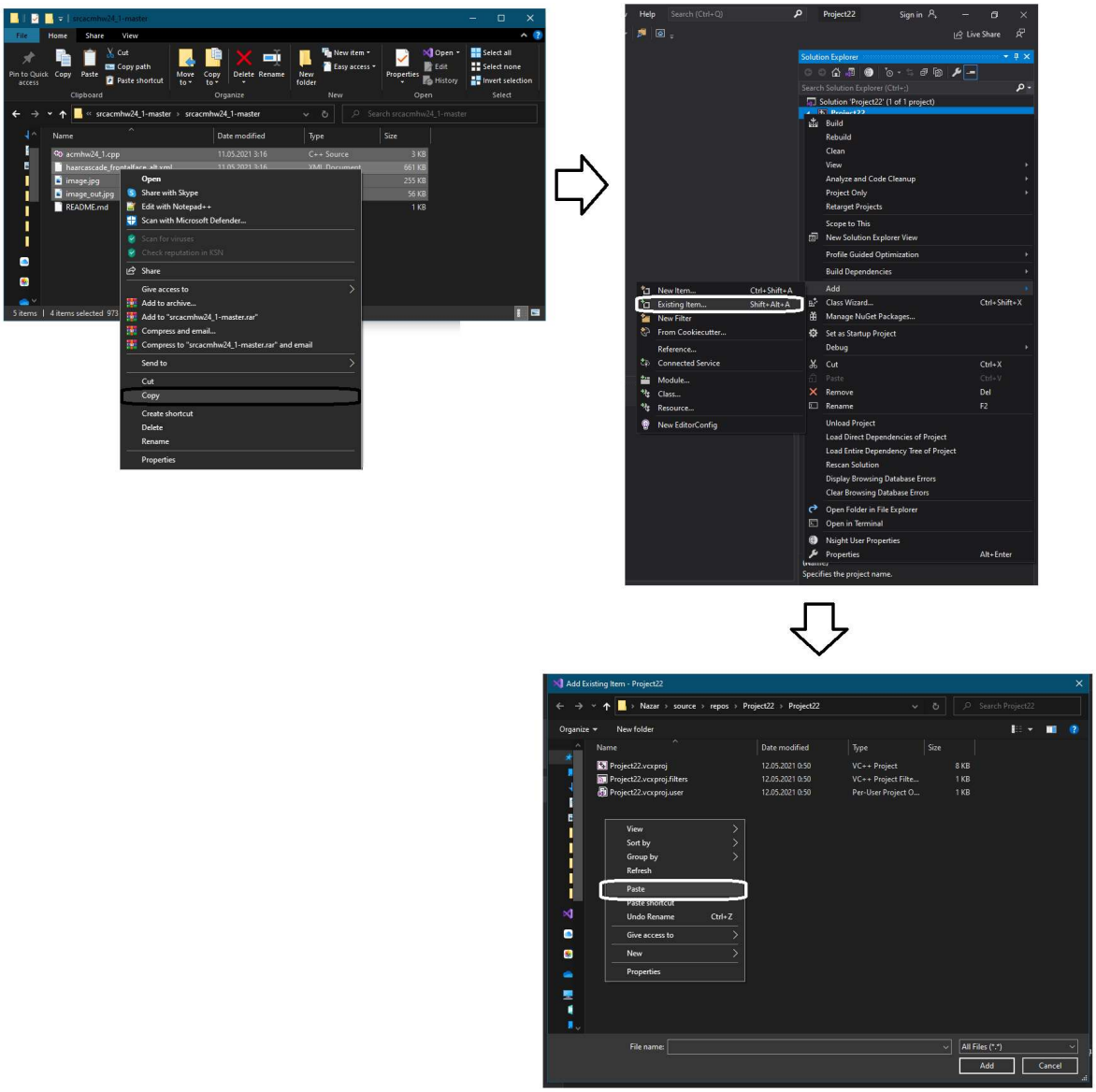

Рис. 3. Копіювання файлів у каталог проекту

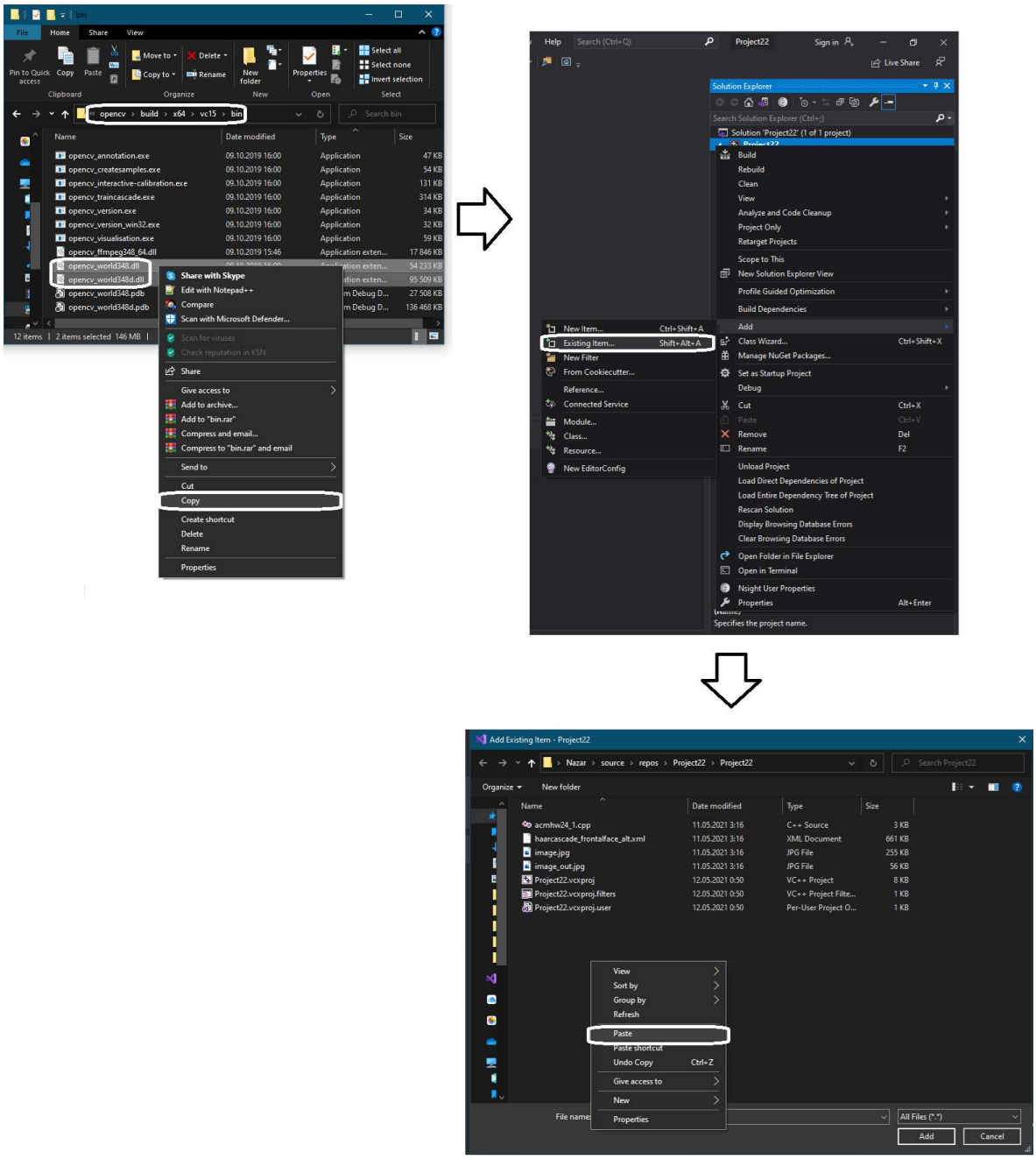

Рис. 4. Копіювання dll-файлів у каталог проекту

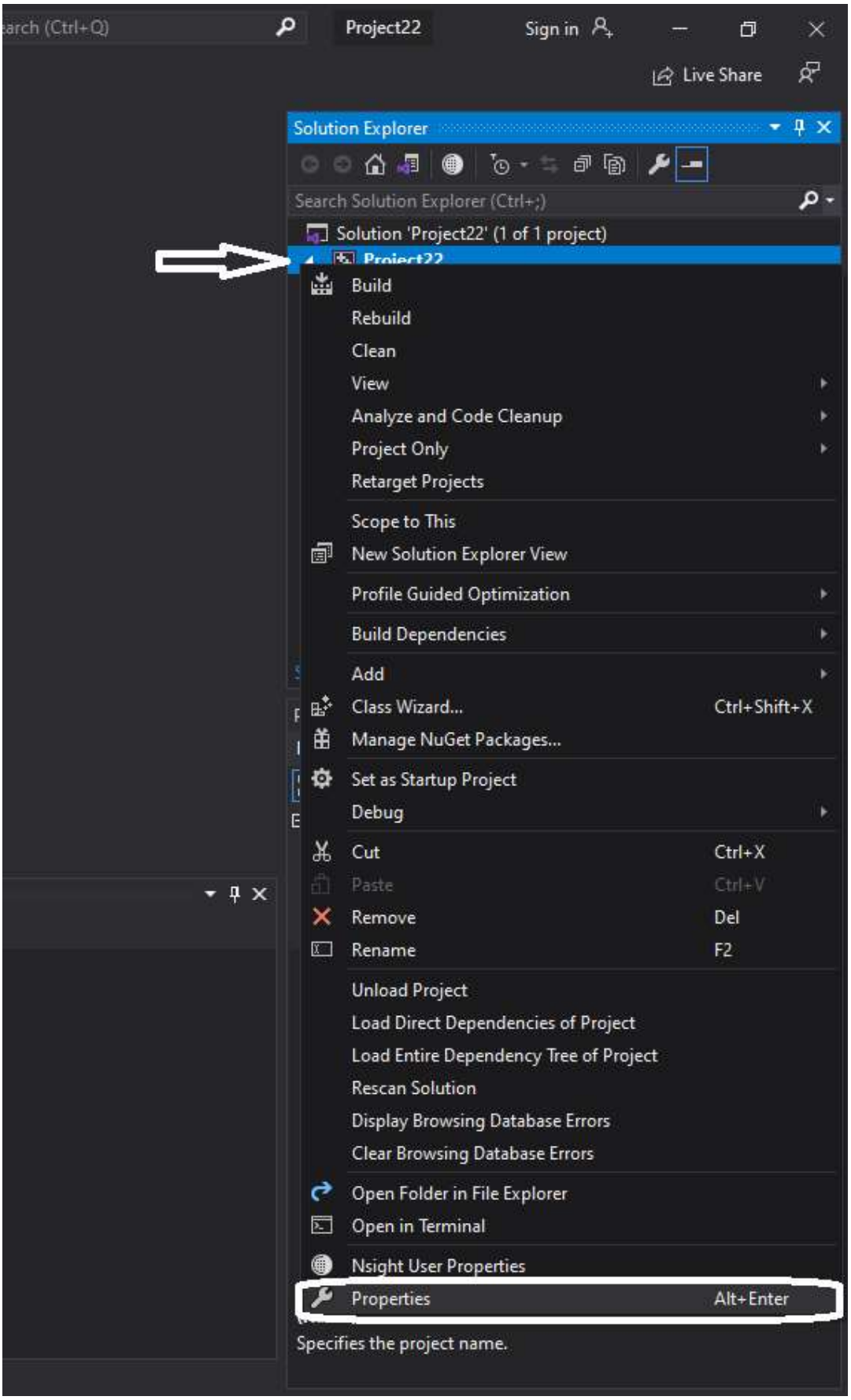

Рис. 5. Перехід до налаштувань

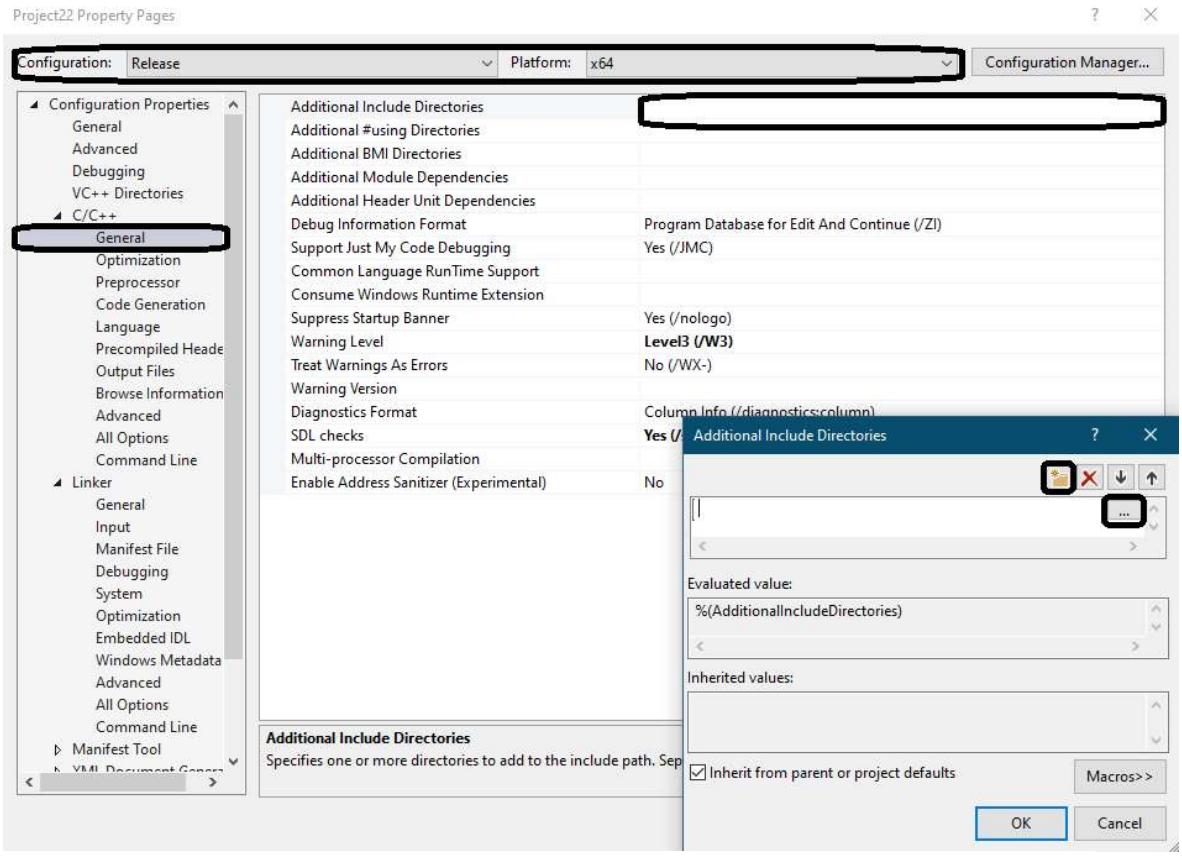

Рис. 6. Налаштування C++

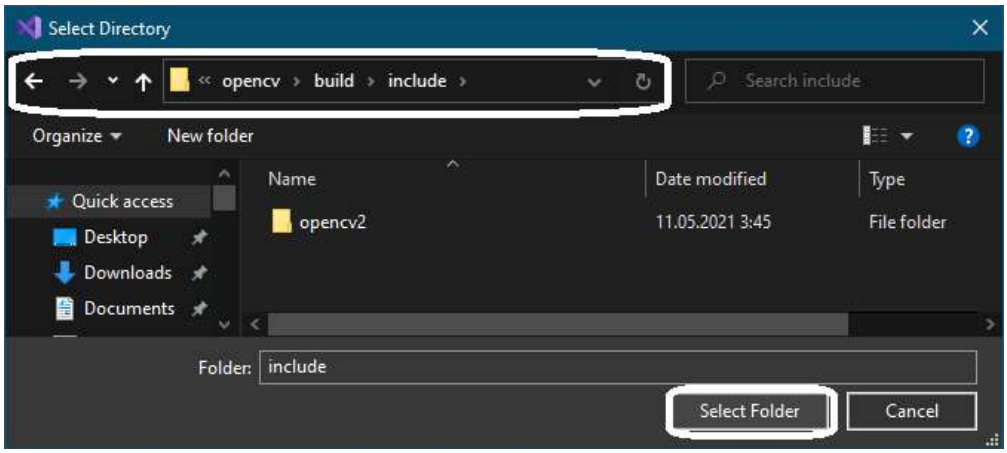

Рис. 7. Вибір каталогу h-файлів

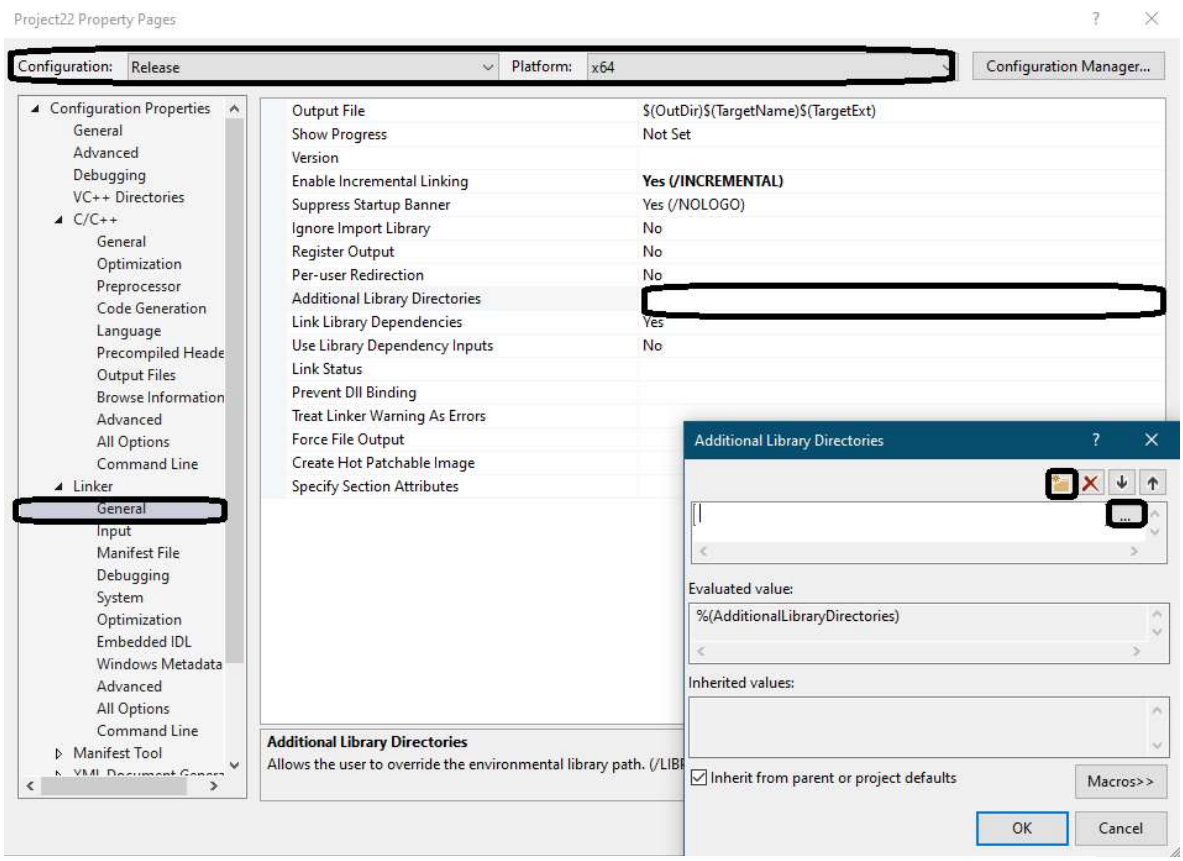

Рис. 8. Налаштування лінкера(компонувальника)

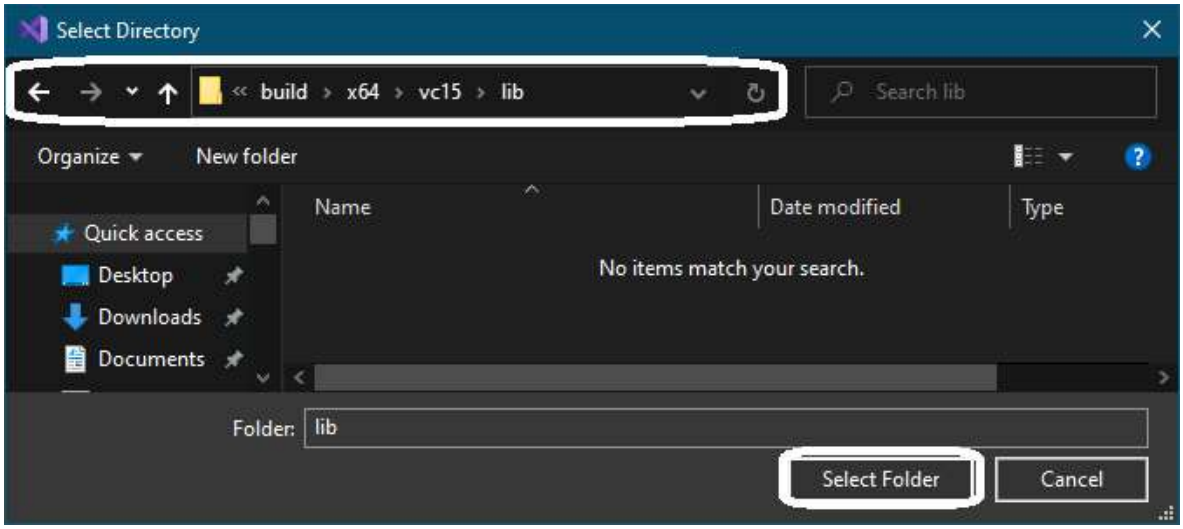

Рис. 9. Вибір каталогу lib-файлів

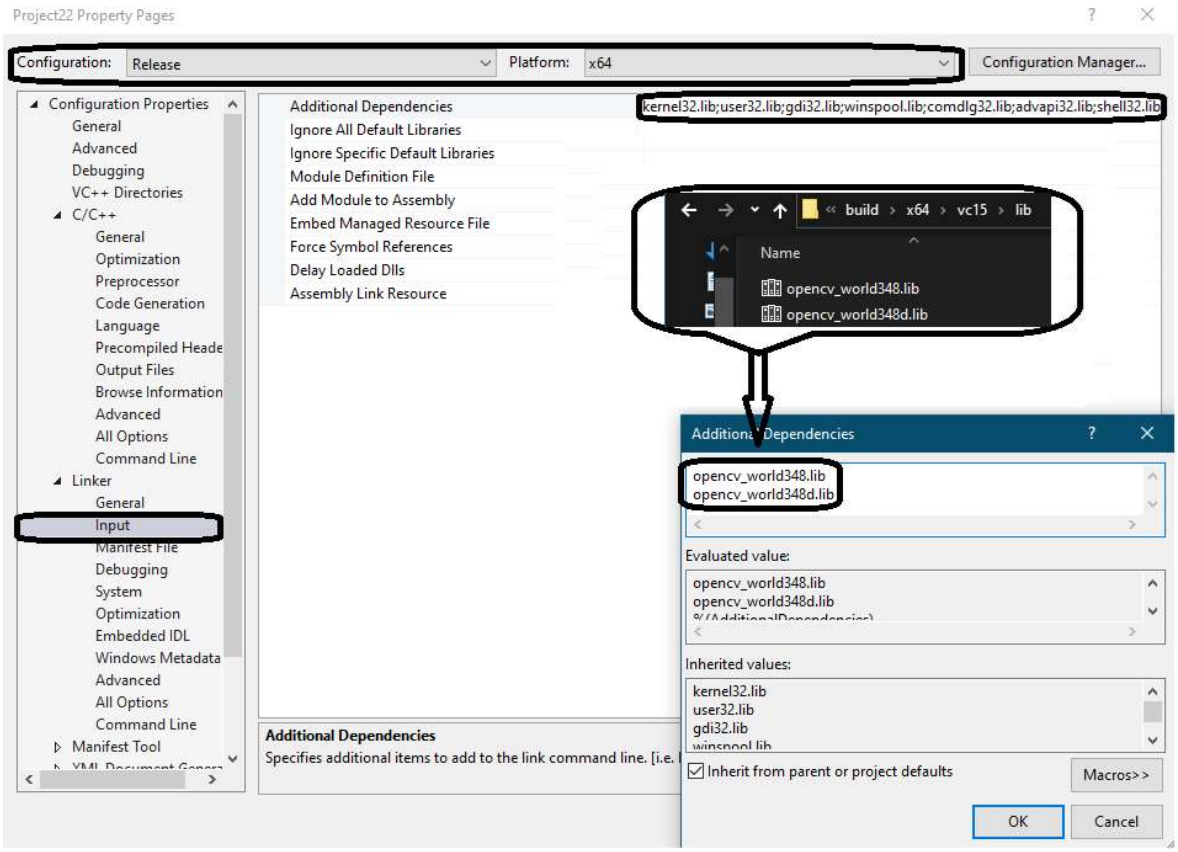

Рис. 10. Означення імен lib-файлів

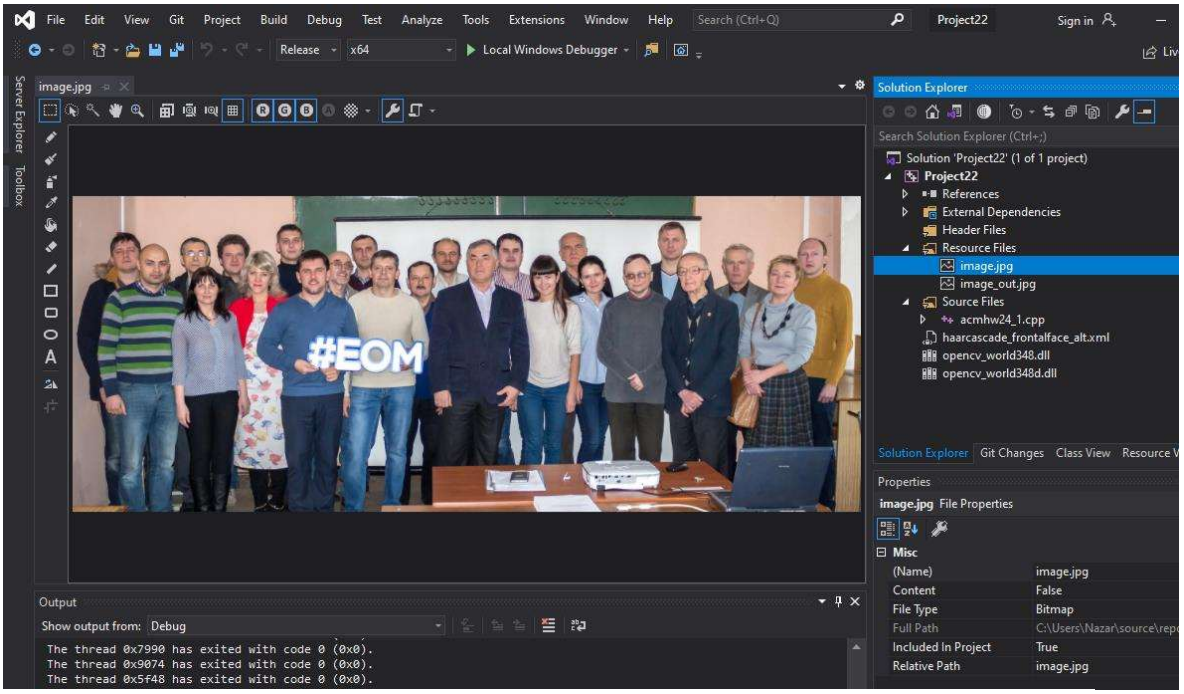

Рис. 11. Відображення оригінального зображення

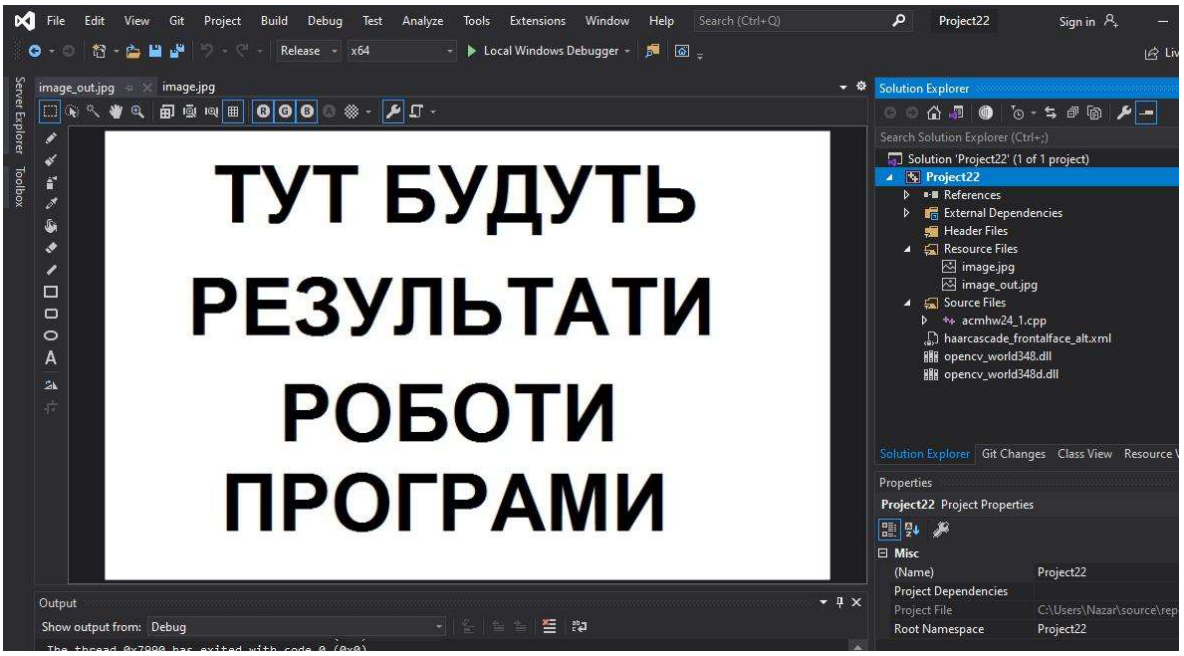

Рис. 12. Файл для збереження результатів виконання програми

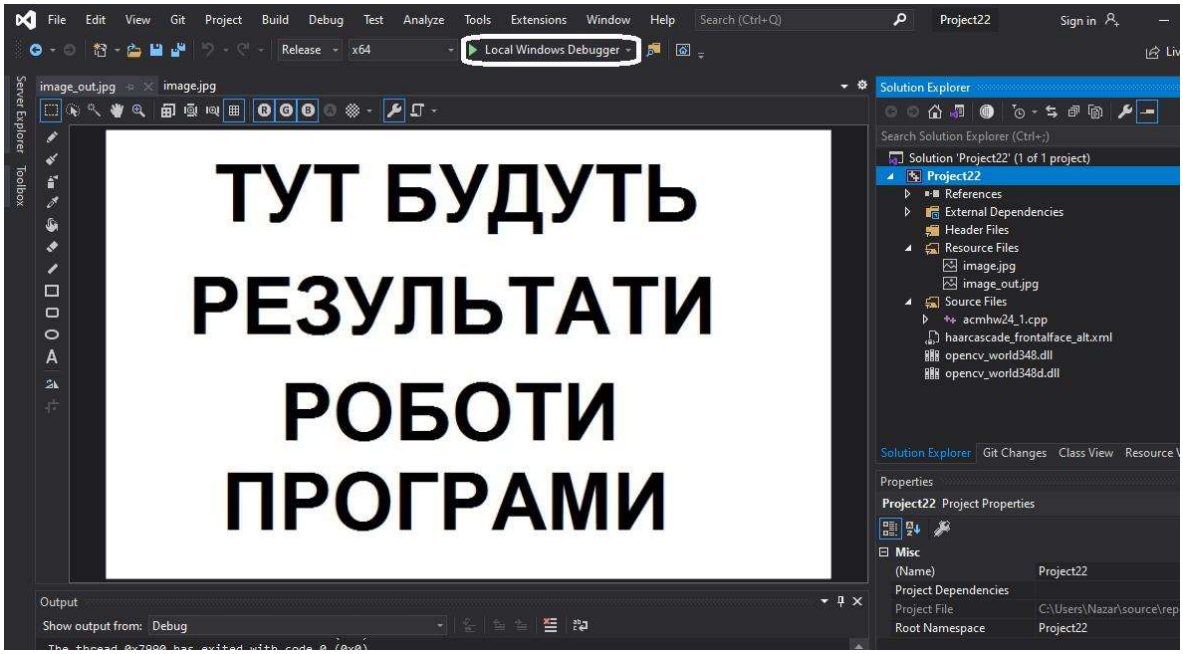

Рис. 13. Компіляція та запуск програми

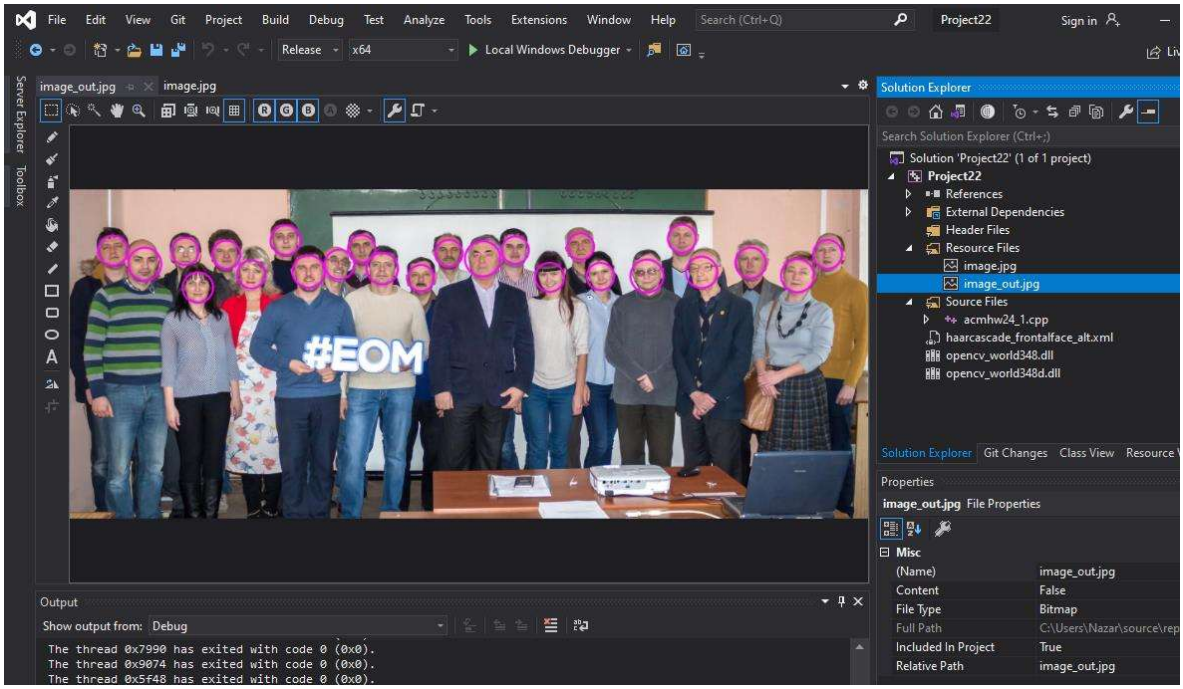

Рис. 14. Оновлене вмістиме файлу для збереження результатів виконання програми

Приклад коду

** до прикладу коду додається файл "haarcascade_frontalface_alt.xml", який знаходиться на репозиторії з кодом, або ж його можна взяти з бібліотеки OpenCV: "opencv/sources/data/haarcascades/haarcascade_frontalface_alt.xml"; цей файл і файли з зображеннями потрібно не забути скопіювати перед виконанням програми*

Наведений зразок коду реалізовує завдання з виконання умови виявлення на фото не більше 24 облич.

Максимальна кількість облич, які можуть бути виявлені наведеним прикладом коду	24
Макровизначення	`#define MAX_FACES_COUNT 24`

Лістинг

```
#include <iostream>
#include <opencv2/opencv.hpp>
//using namespace cv;
#define MAX_FACES_COUNT 24
#define PROCESSING_IMAGE_FROM_FILE

void detectFaces(cv::Mat * image, cv::CascadeClassifier * face_cascade,
std::vector<cv::Rect> * faces) {
    if (!image || !face_cascade || face_cascade->empty() || !faces) {
        return;
    }
    face_cascade->detectMultiScale(*image, *faces, 1.1, 2, 0 | cv::CASCADE_SCALE_IMAGE,
cv::Size(30, 30));
    for (int index = 0; index < faces->size() && index < MAX_FACES_COUNT; ++index) {
        ellipse(
            *image,
            cv::Point((*faces)[index].x + (*faces)[index].width / 2, (*faces)[index].y +
(*faces)[index].height / 2),
            cv::Size((*faces)[index].width / 2, (*faces)[index].height / 2),
            0,
            0,
            360,
            cv::Scalar(255, 0, 255),
            4,
            8,
            0);
    }
}

int main() {
    cv::CascadeClassifier face_cascade;
    if (!face_cascade.load("haarcascade_frontalface_alt.xml")) {
        std::cout << "Could not open haarcascade_frontalface_alt.xml" << std::endl;
        std::cout << "Copy this file and try again" << std::endl;
        return -1;
    }

    cv::Mat image;
    std::vector<cv::Rect> faces;
#ifdef PROCESSING_IMAGE_FROM_FILE
    image = cv::imread("image.jpg");
```

```
    if (!image.data) {
        std::cout << "Could not open or find the image" << std::endl;
        std::cout << "Copy the image file and try again" << std::endl;
        return -1;
    }
    detectFaces(&image, &face_cascade, &faces);

    //cv::imshow("Detected Face", image);
    //cv::waitKey(0);
    cv::imwrite("image_out.jpg", image);
#else
    cv::VideoCapture cap(0);
    while (true) {
        cap >> image;
        detectFaces(&img, &face_cascade, &faces);

        cv::imshow("Detected Face", image);
        cv::waitKey(1);
    }
#endif

#ifdef __linux__
    std::cout << "Press any key to continue . . . " << std::endl;
    (void)getchar();
#elif defined(_WIN32)
    system("pause");
#else
#endif
     return 0;
}
```