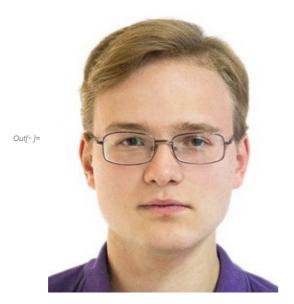
```
In[<sup>®</sup>]:= (*Определение варианта*)
     nNomBrs = 10;
     nNom = (nNomBrs - 1) * 2 + 1
     nNomImage = Mod[nNom, 4] + 1
                 остаток от деления
Out[*]= 19
Out[*]= 4
In[≈]:= (*Постановка задания*)
     (*Размер фотографии: 240x320*)
     (*Размер блока, точек: 4х4*)
     (*Метод встраивания ЦВЗ: метод блочного сокрытия*)
     (*Сего путь: последовательно, непрервно, по столбцам.
     *)
     blockSize = {4, 4};
In[⊕]:= (*Подготовка изображения*)
     image = Import["D:\\GitHub Repos\\stud\\mag\\Sem9\\KP \LT3W\\MY\\240x320.bmp"]
             импорт
                    дифференциировать
```

ImageDimensions[image]

размеры изображения



 $Out[*] = \{240, 320\}$

```
In[*]:= (*Информация для встраивания*)
    rawInfo = "Наименование документа: Зачетная книжка;
    Номер документа: 0020201253;
    Фамилия: Кутузов;
    Имя: Илья;
    Отчество: Геннадьевич;
    Группа: А-12м-20;
    Номер по списку в группе: 10;
    Дата выдачи 20200901;
    Курс: первый.";
In[*]:= (*Подготовка информации для встраиваная*)
    binaryInfo = ToCharacterCode[rawInfo];
                 код символа
    commonLengthInfo = IntegerDigits[binaryInfo, 2, 11];
                       цифры целого числа
    stringInfoList = Flatten[commonLengthInfo];
                     УПЛОСТИТЬ
    stringInfoString = StringJoin[IntegerString[stringInfoList]]
                       соединить с… строковая запись целого числа
```

```
որբեր։ (*Определение цветового канала для встраивания информации*)
     (*Цветовой канал - зеленый*)
     colorChannel = Mod[nNom, 3]
                    остаток от деления
     pixel = {0, 0, 0}; pixel[[colorChannel + 1]] = 255;
     Image[{{pixel}}, "Byte"]
                        байт
Out[=]= 1
Out[@]=
In[*]:= str = {};
     Do [
     оператор цикла
      AppendTo[str, FromCharacterCode[FromDigits[Partition[stringInfoList, 11][[i]], 2]]],
      добавить в коне… символ по его коду
                                       число по ря⋯ разбиение на блоки
      {i, Length[stringInfoList] / 11}]; StringJoin[str]
                                           соединить строки
\mathit{Out}[^{\#}]= Наименование документа: Зачетная книжка;
     Номер документа: 0020201253;
     Фамилия: Кутузов;
     Имя: Илья;
     Отчество: Геннадьевич;
     Группа: А-12м-20;
     Номер по списку в группе: 10;
     Дата выдачи 20200901;
     Курс: первый.
In[*]:= ProtectWatermark[img0_, strInfo0_] := Module[
                                              Іпрограммный модуль
        {img = img0, strInfo = strInfo0, randomSeed = 10, colorChannelCd = 2, blockSize = {4, 4}
        , waterMark
        , imgParts
         , imgPartsColorSeparated
         , storedDimensions
        , imgFlow
        , imgPartsData
        , imgRow
        , separatedColorRow
        , targetChannel
        , workRow
        , parityList
        , parityListToDo
        , randPixelId
        , blocks
        , combinedColorRow
         , newImgParts},
        (*Получение битового списка из входной строки*)
       waterMark = Flatten[IntegerDigits[ToCharacterCode[strInfo], 2, 11]];
```

```
Глиностите Гимфры пеного д Пкод симвона
(*Добавление стоп символа из 11 нулей*)
waterMark = Flatten[Append[waterMark, IntegerDigits[0, 2, 11]]];
            уплостить добавить в конец
                                      цифры целого числа
(*Получение Блоков изображения*)
imgParts = ImagePartition[img, blockSize];
          разбиение изображения
(*Обмен строк со столбцами*)
imgParts = Transpose[imgParts];
          транспозиция
(*Непрерывность пути*)
Do[imgParts[[i]] = Reverse[imgParts[[i]], 1], {i, 2, Length[imgParts], 2}];
                   расположить в обратном порядке
imgRow = Flatten[imgParts];
        уплостить
separatedColorRow = {};
Do[AppendTo[separatedColorRow, ColorSeparate[imgRow[[i]]]], {i, Length[imgRow]}];
                                разделить цветовые каналы
(*Выделение канала в рабочую строку*)
workRow = {};
Do[AppendTo[workRow, ImageData[separatedColorRow[[i, colorChannelCd]], "Byte"]],
_... добавить в конец к
                     данные изображения
 {i, Length[separatedColorRow]}];
     длина
(*Получение четности блоков*)
parityList = {};
Do[AppendTo[parityList, Mod[Total[Flatten[workRow[[i]]]], 2]],
                        ос… сумм… уплостить
 {i, Length[waterMark]}];
(*Наложение сообщения на четность блоков для получения
 списка блоков к изменению четнгости*)
parityListToDo = Mod[parityList + waterMark, 2];
                 остаток от деления
(*Приведение четности блоков к требуемым значениям*)
Do [
_оператор цикла
 randPixelId =
  {RandomInteger[{1, blockSize[[1]]}], RandomInteger[{1, blockSize[[2]]}]};
   случайное целое число
                                         случайное целое число
 workRow[[i, randPixelId[[1]], randPixelId[[2]]]] =
  BitXor[parityListToDo[[i]], workRow[[i, randPixelId[[1]], randPixelId[[2]]]]]],
  сложение битов по модулю 2
 {i, Length[parityListToDo]}
(*Сборка изображения*)
blocks = {};
Do[AppendTo[blocks, Image[workRow[[i]], "Byte"]], {i, Length[workRow]}];
... добавить в конец к изображение
separatedColorRow[[All, colorChannelCd]] = blocks;
```

```
combinedColorRow = {};
Do[AppendTo[combinedColorRow, ColorCombine[separatedColorRow[[i]], "RGB"]],
_... _добавить в конец к
                                комбинировать цвета
  {i, Length[separatedColorRow]}];
 newImgParts = Partition[combinedColorRow, Dimensions[imgParts][[2]]];
              разбиение на блоки
                                            размеры массива
Do[newImgParts[[i]] = Reverse[newImgParts[[i]], 1],
оператор цикла
                       расположить в обратном порядке
  {i, 2, Length[newImgParts], 2}];
        Длина
 newImgParts = Transpose[newImgParts];
              _транспозиция
 ImageAssemble[newImgParts]
собрать изображение
]
```

```
In[@]:= ReadWatermark[img0_] := Module[
                              программный модуль
       {img = img0, colorChannelCd = 2, blockSize = {4, 4}, randomSeed
        , strInfo
        , waterMark
        , imgParts
        , imgPartsData
        , imgRow
        , workRow
       },
       (∗Получение битового списка из входной строки∗)
       imgParts = ImagePartition[img, blockSize];
                 разбиение изображения
       imgParts = Transpose[imgParts];
                 транспозиция
       Do[imgParts[[i]] = Reverse[imgParts[[i]], 1], {i, 2, Length[imgParts], 2}];
                          расположить в обратном порядке
      оператор цикла
                                                              длина
       imgRow = Flatten[imgParts];
               уплостить
      workRow = {};
      Do[AppendTo[workRow, ImageData[
      ______ добавить в конец к _______ данные изображения
          ColorSeparate[imgRow[[i]]][[colorChannelCd]], "Byte"]], {i, Length[imgRow]}];
          разделить цветовые ка…
                                                             байт
       waterMark = {};
      Do[AppendTo[waterMark, Mod[Total[Flatten[workRow[[i]]]], 2]],
      ... добавить в конец к
                               ос… сумм… уплостить
        {i, Length[workRow]}];
            длина
       waterMark = Partition[waterMark, 11];
                  разбиение на блоки
       waterMark =
        Take [waterMark, Flatten [Position [waterMark, IntegerDigits [0, 2, 11]]] [[1]] - 1];
                        уплостить позиция по образцу
                                                      цифры целого числа
       strInfo = {};
       Do[AppendTo[strInfo, FromDigits[waterMark[[i]], 2]], {i, Length[waterMark]}];
         добавить в конец к
                            число по ряду цифр
       FromCharacterCode[strInfo]
      символ по его коду
      ]
```

In[*]:= protectedImage = ProtectWatermark[image, rawInfo] filePath =

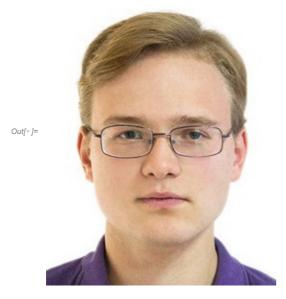
Export["D:\\GitHub Repos\\stud\\mag\\Sem9\\KP LT3M\\MY\\protected_image.bmp",

_экспорт… _дифференциировать

protectedImage];

ReadWatermark[Import[filePath]]

импорт



Out[=]= Наименование документа: Зачетная книжка;

Номер документа: 0020201253;

Фамилия: Кутузов;

Имя: Илья;

Отчество: Геннадьевич;

Группа: А-12м-20;

Номер по списку в группе: 10;

Дата выдачи 20200901;

Курс: первый.

```
ln[+]:= p = 13642163;
     a = 0;
     b = 23;
     c = 66;
     P = \{6821082, 5569902\};
     rank = 13645001;
     secretKey = RandomInteger[{1, rank - 1}]
                 случайное целое число
     publicKey = EllipticMult[p, a, b, c, P, secretKey]
     signature = ECDSAGeneration[p, a, b, c + 59, P, rank, filePath, secretKey]
     ECDSAVerification[p, a, b, c + 59, P, rank, filePath, publicKey, signature]
     ECDSAVerification[p, a, b, c + 59, P, rank,
      "D:\\GitHub Repos\\stud\\mag\\Sem9\\KP \LT3M\\MY\\240x320.bmp", publicKey, signature]
       дифференциировать
Out[*]= 3 181 056
Out[\#] = \{6030880, 868506\}
Out[*]= {13 177 322, 6 231 937}
     True
     False
```