Code based on python and GDAL

```
# Import package
import gdal
import numpy as np
import random


# Slicing operation
def make_slices(data, win_size):
    rows = data.shape[0] - win_size[0] + 1
    cols = data.shape[1] - win_size[1] + 1
    slices = []
    for i in range(win_size[0]):
        for j in range(win_size[1]):
            slices.append(data[i:rows + i, j:cols + j])
    return slices


# Create a one-band GeoTIFF
def make_raster(in_ds, fn, data, data_type, nodata=None):
    driver = gdal.GetDriverByName('GTiff')
    out_ds = driver.Create(
        fn, in_ds.RasterXSize, in_ds.RasterYSize, 1, data_type)
    out_ds.SetProjection(in_ds.GetProjection())
    out_ds.SetGeoTransform(in_ds.GetGeoTransform())
    out_band = out_ds.GetRasterBand(1)
    if nodata is not None:
        out_band.SetNoDataValue(nodata)
    out_band.WriteArray(data)
    out_band.FlushCache()
    out_band.ComputeStatistics(False)
    return out_ds


# Data preparation
def datapre(in_raster, in_ref):
    data = gdal.Open(in_raster)
    ref = gdal.Open(in_ref)
    band = data.GetRasterBand(1)
    dtype = band.ReadAsArray().astype(np.int32)
    return data, ref, band, dtype
```

```
# Select pure pixels
def choice(band, dtype, num):
    # Read the number of pixels in the X direction and the number of pixels in the Y direction of the grid dataset
    Rows, Cols = band.YSize, band.XSize
    # Set window size
    rdata = make_slices(dtype, (3, 3))
    stacked_data = np.ma.dstack(rdata)
    tdata = np.ones((Rows, Cols), np.int32) * -99
    tdata[1:-1, 1:-1] = np.mean(stacked_data, 2)
    A = tdata[1:-1, 1:-1]
    l = A.shape[0]
    h = A.shape[1]
    tarray = []
    for ii in np.arange(0, l):
        for jj in np.arange(0, h):
            # 1-water;2-non water
            if A[ii, jj] == 1:
                # Add "1" to make the size consistent with the original image
                tarray.append([ii + 1, jj + 1, 1])
    # Randomly selected samples
    tarraych = random.sample(tarray, num)
    # nodata -> -99
    tarrayr = np.ones((Rows, Cols), np.int32) * -99
    for li in tarraych:
        tarrayr[li[0]][li[1]] = li[2]
    return tarrayr


# assembly
def all(in_raster, in_ref, out_raster, num):
    data, ref, band, dtype = datapre(in_raster, in_ref)
    outarray = choice(band, dtype, num)
    make_raster(ref, out_raster, outarray, gdal.GDT_Int32, -99)


# Example
if __name__ == '__main__':
    # data
    in_raster = r"D:\L1Ctiff\learnAre\ZXWater.tif"
    # Data space reference
    in_ref = r"D:\L1Ctiff\learnAre\ZXWater.tif"
    # Write and store output
```

```
out_raster = r"D:\L1Ctiff\learnAre\ZXWater10086.tif"

all(in_raster, in_ref, out_raster, 2000)
```