```
#  d2 zadanie 7

# znajduje obszar wokol punktu o wysokosci h_pkt do przewyzszenia D+h_mean


from PyQt5.QtGui import *
from PyQt5.QtCore import *
from qgis.core import *
from qgis.utils import *
import processing

otoczenie =25 #otoczenie punktu wybranego w m
D = 10 # przewyzszenie w m ponad srednia wys w otoczeniu
id_pkt = "2"

workPath = "B:\\Python_QGIS\\aQGIS_Hel_2018\\Jacka\\moje_J\\D2_J_AW\\"
dem_in = QgsRasterLayer('B:/Python_QGIS/aQGIS_Hel_2018/Jacka/dane2_18/dem_Krak.tif', "dem_in")
if not dem_in.isValid():
    print("Layer 2 failed to load!")
pkt_in =
QgsVectorLayer("B:/Python_QGIS/aQGIS_Hel_2018/Jacka/dane2_18/pl_stacje_pom_Krak.shp",
"punkty_in", "ogr")
if not pkt_in.isValid():
    print("Layer 1 failed to load!")

exp_pkt="IDPP = "+id_pkt

pkt_in.setSubsetString(exp_pkt)  #ograniczenie w-wy punktow do wybranego punktu

#odczyt wysokości wybranego punktu - gdy za dolną granicę przyjmowałam średnią z otoczenia
to nie zawsze wybrany punkt znajdował się wewnątrz utworzonego poligonu
feat_pkt = pkt_in.getFeatures()
for feat in feat_pkt:
    geomF = feat.geometry()
    pkt= geomF.asPoint()  # obiekt  QgsPointXY
h_pkt_i = dem_in.dataProvider().identify(pkt,QgsRaster.IdentifyFormatValue)  #dostęp do
wartości rastra w punkcie
if h_pkt_i.isValid():
    h_pkt = h_pkt_i.results()[1]   #odczyt wysokosci w wybranym punkcie ze słownika będącego
    rezultatem QgsRaster.IdentifyFormatValue
    print(h_pkt)
        #print(h_pkt_i.results())
#buforowanie
buf_1 = workPath + "buf_pkt.shp"
processing.run("native:buffer",{'INPUT':pkt_in,
'DISTANCE':otoczenie,'SEGMENTS':5,'END_CAP_STYLE':0,'JOIN_STYLE':0,'MITER_LIMIT':2,'DISSOLVE':
False,'OUTPUT':buf_1})

#zonal statistics
#zonal stat nie tworzzy nowej warstwy tylko dopisuje statystyki do warstwy stref
#by odczytac wartosc z tabeli dodaje warstwe do projektu, a potem ja usuwam z projektu by
mozna bylo znow puscic program
processing.run("qgis:zonalstatistics",
{'INPUT_RASTER':dem_in,'RASTER_BAND':1,'INPUT_VECTOR':buf_1,'COLUMN_PREFIX':'ZS_','STATS':[0,1
,2]})
buf_pkt1 = iface.addVectorLayer(buf_1,"buf_pkt","ogr")  #dodanie do projektu
ZS_mean = buf_pkt1.getFeature(0) #dostęp do obiektu - buf_pkt to warstwa z jednym obiektem
h_mean = ZS_mean['ZS_mean'] #dostęp do wartości w atrybucie
QgsProject.instance().removeMapLayer(buf_pkt1)  #usunięcie warstwy z projektu

h_max = h_mean + D

#formuła do reklasyfikacji DEM_in
form = "where(A >= "+str(h_pkt)+", where(A <= "+str(h_max)+",1,65535),65535)"
#print(form)

#reklasyfikacja
```

```
ras_1 = workPath + "R1_dem_h_D.tif"
processing.run("gdal:rastercalculator",
{'INPUT_A':dem_in,'BAND_A':1,'INPUT_B':None,'BAND_B':-1,'INPUT_C':None,'BAND_C':-1,'INPUT_D':None,'BAND_D':-1,'INPUT_E':None,'BAND_E':-1,'INPUT_F':None,'BAND_F':-1,'FORMULA':form,'NO_DATA':65535,'RTYPE':5,'EXTRA':'','OPTIONS':'','OUTPUT':ras_1})

#region group - nadanie identyfikatorów każdej połączonej grupie pikseli
ras_2 = workPath + "R2_RG_reclass.tif"
processing.run("grass7:r.clump",
{'input':ras_1,'title':'RG','-d':False,'output':ras_2,'GRASS_REGION_PARAMETER':None,'GRASS_REGION_CELLSIZE_PARAMETER':0,'GRASS_RASTER_FORMAT_OPT':'','GRASS_RASTER_FORMAT_META':''})

#sczytanie numeru grupy na której znajduje się punkt
ras_RG_lyr = iface.addRasterLayer(ras_2,"rasRG")
nr_RG_i = ras_RG_lyr.dataProvider().identify(pkt,QgsRaster.IdentifyFormatValue)
if nr_RG_i.isValid():
    nr_RG = nr_RG_i.results()[1]   #odczyt wysokosci w wybranym punkcie
    int_RG=int(nr_RG)
QgsProject.instance().removeMapLayer(ras_RG_lyr)

#formuła do reklasyfikacji mapy grup - wybranie grupy na której znajduje się wybrany punkt
ff = 'where(A == '+str(int_RG)+',1,65535)'

#reklasyfikacja by dostać raster z jedną grupą
ras_3 = workPath + "R3_poligon_ok.tif"
processing.run("gdal:rastercalculator",
{'INPUT_A':ras_2,'BAND_A':1,'INPUT_B':None,'BAND_B':-1,'INPUT_C':None,'BAND_C':-1,'INPUT_D':None,'BAND_D':-1,'INPUT_E':None,'BAND_E':-1,'INPUT_F':None,'BAND_F':-1,'FORMULA':ff,'NO_DATA':65535,'RTYPE':5,'EXTRA':'','OPTIONS':'','OUTPUT':ras_3})

#raster to polygon
poly_out = workPath +"poly_pkt_"+id_pkt+".shp"
processing.run("gdal:polygonize",
{'INPUT':ras_3,'BAND':1,'FIELD':'DN','EIGHT_CONNECTEDNESS':False,'OUTPUT':poly_out})

pkt_in.setSubsetString('')

print("OK")
```