Supplementary materials

Supplementary material I Construction and Demolition in Leiden for the period 2020–2030

Table S1. Planned demolition in m² for the municipality of Leiden, 2020–2030.

	Row House	Offices	High Rise	Commercial	Other	Detached	Apartment
2019	0	1845	52,613	0	22,706	1831	14,511
2020	4096	1590	52,114	0	3796	1298	48,733
2021	12,082	1321	978	15,124	32,317	0	2340
2022	0	41,122	35,755	0	50,443	0	2725
2023	0	6115	0	2747	48,875	0	0
2024	0	0	0	0	13,468	0	0
2025	0	18,249	0	0	31,542	97	16,599
2026	2311	10,035	20,209	2553	29,021	461	12,130
2027	2311	10,035	20,209	2553	29,021	461	12,130
2028	2311	10,035	20,209	2553	29,021	461	12,130
2029	0	0	0	0	11,925	0	0
2030	0	0	0	0	35,526	0	0

Table S2. Planned construction in m² for the municipality of Leiden, 2020–2030.

	Row House	Offices	High Rise	Commercial	Other	Detached	Apartment
2019	0	0	23,080	0	25,890	1640	11,760
2020	7600	1440	60,910	0	5860	1760	32,400
2021	13,120	7200	3520	10,240	95,920	0	3280
2022	0	68,160	26,300	0	75,560	0	0
2023	0	52,920	0	4000	95,880	0	0
2024	0	0	0	0	48,300	0	0
2025	0	40,000	0	0	32,300	49,280	32,000
2026	2960	24,246	16,259	2034	54,244	7526	11,349
2027	2960	24,246	16,259	2034	54,244	7526	11,349
2028	2960	24,246	16,259	2034	54,244	7526	11,349
2029	0	0	0	0	65,160	0	0
2030	0	0	0	0	100,000	0	0

Total demolition: 814,640 m²; Total construction: 1,351,331 m².

Supplementary material II Construction and Demolition Waste in Leiden

Table S3. Demolition waste supply per year (tonnes).

	Concrete	Brick	Wood	Roof Gravel	Aluminium	Steel	Glass	Ceramic	Gypsum	Bitumen	Cast Iron	Other Materials
2019	80,838	26,734	5539	1269	642	1133	1931	1572	1087	984	338	347
2020	119,258	41,645	7229	278	700	567	2222	2282	789	1468	521	154
2021	44,605	11,570	3000	2520	531	1853	1431	541	1518	1063	75	587
2022	97,802	21,724	5513	4731	1054	3490	3183	1219	2669	1595	189	1109
2023	43,104	9725	2084	2983	516	2118	1373	254	1650	796	-	678
2024	10,112	2474	481	696	120	493	309	59	393	171	-	158
2025	62,330	15,585	2834	2573	548	1888	1615	575	1475	1028	82	600
2026	65,436	18,494	3811	2150	587	1649	1723	929	1369	1015	172	519
2027	65,436	18,494	3811	2150	587	1649	1723	929	1369	1015	172	519

2028	65,436	18,494	3811	2150	587	1649	1723	929	1369	1015	172	519
2029	8954	2190	426	616	107	436	273	52	348	151	-	140
2030	26,675	6525	1268	1836	317	1300	815	156	1036	451	-	417

Table S4. Construction material demand per year (tonnes).

	Concrete	Brick	Wood	Roof Gravel	Aluminium	Steel	Glass	Ceramic	Gypsum	Bitumen	Cast Iron	Other Materials
2019	55,949	18,063	3537	1338	455	1073	1310	889	964	740	179	335
2020	103,818	36,222	7109	377	695	619	2194	2212	813	1272	503	173
2021	96,705	23,948	5495	5857	1135	4228	3048	921	3408	1875	98	1349
2022	125,109	23,653	6513	7426	1446	5365	4355	1190	3990	2230	129	1711
2023	113,084	20,658	5330	7895	1366	5611	3925	672	4190	2180	0	1796
2024	36,266	8871	1724	2496	431	1767	1108	213	1409	613	0	567
2025	152,889	57,208	16,781	3736	1147	2978	3572	2050	2462	2309	398	921
2026	97,688	26,946	6641	4161	954	3092	2787	1164	2462	1603	187	979
2027	97,688	26,946	6641	4161	954	3092	2787	1164	2462	1603	187	979
2028	97,688	26,946	6641	4161	954	3092	2787	1164	2462	1603	187	979
2029	48,925	11,968	2326	3367	582	2384	1494	287	1901	827	0	765
2030	75,085	18,367	3570	5167	893	3659	2293	440	2917	1269	0	1175

Table S5. Material supply from demolition and demand from construction per year (tonnes).

		Concrete	Brick	Wood	Roof Gravel	Aluminium	Steel	Glass	Ceramic	Gypsum	Bitumen	Cast Iron	Other Materials
2019	Supply	80,838	26,734	5539	1269	642	1133	1931	1572	1087	984	338	347
	Demand	55,949	18,063	3537	1338	455	1073	1310	889	964	740	179	335
	Balance	24,889	8670	2002	−69	187	60	621	683	123	245	159	12
2020	Supply	119,258	41,645	7229	278	700	567	2222	2282	789	1468	521	154
	Demand	103,818	36,222	7109	377	695	619	2194	2212	813	1272	503	173
	Balance	15,440	5424	120	−99	5	−52	27	69	−24	196	17	−19
2021	Supply	44,605	11,570	3000	2520	531	1853	1431	541	1518	1063	75	587
	Demand	96,705	23,948	5495	5857	1135	4228	3048	921	3408	1875	98	1349
	Balance	−52,100	−12,378	−2495	−3338	−605	−2375	−1617	−380	−1890	−812	−22	−762
2022	Supply	97,802	21,724	5513	4731	1054	3490	3183	1219	2669	1595	189	1109
	Demand	125,109	23,653	6513	7426	1446	5365	4355	1190	3990	2230	129	1711
	Balance	−27,307	−1929	−1000	−2695	−391	−1875	−1172	29	−1321	−635	60	−603
2023	Supply	43,104	9725	2084	2983	516	2118	1373	254	1650	796	-	678
	Demand	113,084	20,658	5330	7895	1366	5611	3925	672	4190	2180	0	1796
	Balance	−69,980	−10,933	−3246	−4912	−850	−3493	−2553	−418	−2540	−1384	-	−1117
2024	Supply	10,112	2474	481	696	120	493	309	59	393	171	-	158
	Demand	36,266	8871	1724	2496	431	1767	1108	213	1409	613	0	567
	Balance	−26,154	−6398	−1244	−1800	−311	−1275	−799	−153	−1016	−442	-	−409
2025	Supply	62,330	15,585	2834	2573	548	1888	1615	575	1475	1028	82	600
	Demand	152,889	57,208	16,781	3736	1147	2978	3572	2050	2462	2309	398	921
	Balance	−90,559	−41,623	−13,947	−1163	−599	−1089	−1957	−1475	−987	−1282	−316	−322
2026	Supply	65,436	18,494	3811	2150	587	1649	1723	929	1369	1015	172	519
	Demand	97,688	26,946	6641	4161	954	3092	2787	1164	2462	1603	187	979
	Balance	−32,253	−8452	−2830	−2011	−366	−1443	−1064	−235	−1094	−588	−15	−460
2027	Supply	65,436	18,494	3811	2150	587	1649	1723	929	1369	1015	172	519
	Demand	97,688	26,946	6641	4161	954	3092	2787	1164	2462	1603	187	979
	Balance	−32,253	−8452	−2830	−2011	−366	−1443	−1064	−235	−1094	−588	−15	−460
2028	Supply	65,436	18,494	3811	2150	587	1649	1723	929	1369	1015	172	519
	Demand	97,688	26,946	6641	4161	954	3092	2787	1164	2462	1603	187	979
	Balance	−32,253	−8452	−2830	−2011	−366	−1443	−1064	−235	−1094	−588	−15	−460
2029	Supply	8954	2190	426	616	107	436	273	52	348	151	-	140
	Demand	48,925	11,968	2326	3367	582	2384	1494	287	1901	827	0	765
	Balance	−39,971	−9778	−1901	−2751	−476	−1948	−1221	−234	−1553	−676	-	−625
2030	Supply	26,675	6525	1268	1836	317	1300	815	156	1036	451	-	417
	Demand	75,085	18,367	3570	5167	893	3659	2293	440	2917	1269	0	1175
	Balance	−48,410	−11,842	−2302	−3331	−576	−2359	−1478	−284	−1881	−818	-	−757

Supplementary material III, Annual Collected and Recycled Demolition Waste in Leiden

Table S6. Material recycling per material and year (tonnes, extrapolated values in dark blue).

		Concrete	Brick	Wood	Roof Gravel	Aluminium	Steel	Glass	Ceramic	Gypsum	Bitumen	Cast Iron	Other Materials
	Collection rate (%)	85%	95%	95%	0%	95%	95%	95%	95%	95%	50%	95%	100%
	Recycled content (%)	50%	50%	90%	0%	50%	85%	91%	80%	40%	50%	96%	0%
2019	Supply	68,712	25,397	5262	-	610	1077	1835	1493	1032	492	321	347
	Demand	55,949	18,063	3537	1338	455	1073	1310	889	964	740	179	335
	Recycled content limit	27,974	9032	3183	-	228	912	1192	711	386	370	172	-
	Recycled	27,974	9032	3183	-	228	912	1192	711	386	370	172	-
2020	Supply	101,369	39,563	6867	-	665	538	2111	2167	750	734	495	154
	Demand	103,818	36,222	7109	377	695	619	2194	2212	813	1272	503	173
	Recycled content limit	51,909	18,111	6398	-	348	526	1997	1770	325	636	483	-
	Recycled	51,909	18,111	6398	-	348	526	1997	1770	325	636	483	-
2021	Supply	37,914	10,991	2850	-	504	1761	1359	514	1442	532	72	587
	Demand	96,705	23,948	5495	5857	1135	4228	3048	921	3408	1875	98	1349
	Recycled content limit	48,352	11,974	4945	-	568	3594	2773	737	1363	937	94	-
	Recycled	37,914	10,991	2850	-	504	1761	1359	514	1363	532	72	-
2022	Supply	83,132	20,637	5238	-	1002	3315	3024	1158	2536	797	179	1109
	Demand	125,109	23,653	6513	7426	1446	5365	4355	1190	3990	2230	129	1711
	Recycled content limit	62,554	11,826	5862	-	723	4560	3963	952	1596	1115	124	-
	Recycled	62,554	11,826	5238	-	723	3315	3024	952	1596	797	124	-
2023	Supply	36,638	9239	1980	-	490	2012	1304	241	1568	398	-	678
	Demand	113,084	20,658	5330	7895	1366	5611	3925	672	4190	2180	-	1796
	Recycled content limit	56,542	10,329	4797	-	683	4769	3572	538	1676	1090	-	-
	Recycled	36,638	9239	1980	-	490	2012	1304	241	1568	398	-	-
2024	Supply	8596	2350	457	-	114	468	293	56	373	85	-	158
	Demand	36,266	8871	1724	2496	431	1767	1108	213	1409	613	-	567
	Recycled content limit	18,133	4436	1552	-	216	1502	1008	170	564	306	-	-
	Recycled	8596	2350	457	-	114	468	293	56	373	85	-	-
2025	Supply	52,980	14,806	2693	-	520	1794	1534	546	1402	514	78	600
	Demand	152,889	57,208	16,781	3736	1147	2978	3572	2050	2462	2309	398	921
	Recycled content limit	76,444	28,604	15,103	-	573	2531	3251	1640	985	1155	382	-
	Recycled	52,980	14,806	2693	-	520	1794	1534	546	985	514	78	-
2026	Supply	55,620	17,569	3621	-	558	1566	1637	882	1300	508	163	519
	Demand	97,688	26,946	6641	4161	954	3092	2787	1164	2462	1603	187	979
	Recycled content limit	48,844	13,473	5977	-	477	2628	2537	931	985	801	179	-
	Recycled	48,844	13,473	3621	-	477	1566	1637	882	985	508	163	-
2027	Supply	55,620	17,569	3621	-	558	1566	1637	882	1300	508	163	519
	Demand	97,688	26,946	6641	4161	954	3092	2787	1164	2462	1603	187	979

	Recycled content limit	48,844	13,473	5977	-	477	2628	2537	931	985	801	179	-
	Recycled	48,844	13,473	3621	-	477	1566	1637	882	985	508	163	-
2028	Supply	55,620	17,569	3621	-	558	1566	1637	882	1300	508	163	519
	Demand	97,688	26,946	6641	4161	954	3092	2787	1164	2462	1603	187	979
	Recycled content limit	48,844	13,473	5977	-	477	2628	2537	931	985	801	179	-
	Recycled	48,844	13,473	3621	-	477	1566	1637	882	985	508	163	-
2029	Supply	7611	2081	404	-	101	415	260	50	330	76	-	140
	Demand	48,925	11,968	2326	3367	582	2384	1494	287	1901	827	-	765
	Recycled content limit	24,463	5984	2094	-	291	2027	1360	229	760	413	-	-
	Recycled	7611	2081	404	-	101	415	260	50	330	76	-	-
2030	Supply	22,673	6199	1205	-	301	1235	774	148	984	225	-	417
	Demand	75,085	18,367	3570	5167	893	3659	2293	440	2917	1269	-	1175
	Recycled content limit	37,542	9183	3213	-	447	3110	2087	352	1167	635	-	-
	Recycled	22,673	6199	1205	-	301	1235	774	148	984	225	-	-

Table S7. Summarized recycling of materials (tonnes).

	Concrete	Brick	Wood	Roof Gravel	Aluminium	Steel	Glass	Ceramic	Gypsum	Bitumen	Cast Iron	Other Materials
Supply	586,487	183,970	37,817	-	5983	17,315	17,405	9022	14,318	5377	1634	5747
Demand	1,100,894	299,796	72,309	50,141	11,011	36,960	31,661	12,366	29,441	18,122	1867	11,731
Recycled	455,383	125,053	35,269	-	4760	17,138	16,648	7636	10,865	5156	1418	-

Supply of demolition waste: 1,033,029 metric tonnes.
Supply of demolition waste after collection: 885,074 metric tonnes.
Demand: 1,676,298 metric tonnes.
Recycled: 679,327 metric tonnes.
66% of demolition waste recycled as secondary materials, 41% lower primary material demand.
14% material of demolition waste not suitable for collection, 20% mismatch.

Supplementary material IV, Python Script

```
# -*- coding: utf-8 -*-
"""
Created on Mon Mar 30 10:34:08 2020

Building stock model v2

@author: Teun

"""

#Import packages
import geopandas as gpd
import pandas as pd
import numpy as np
#import matplotlib.pyplot as plt

#%% Data import

#Read shapefile and set it as dataframe df
```

```
df = gpd.read_file(r'C:\Users\Teun\Documents\Spatial data\Leiden\Leiden\Leiden_woningtypes.shp')


#Input of building material data
data = pd.ExcelFile(r'C:\Users\Teun\Google Drive\TB business\New Horizon business\New Horizon data & building typology\Code\Materials_Mit.xlsx')

#Selection of excel sheets per building type
df_Mi_Residential = pd.read_excel(data,'Residential') #Dit nog aanvullen naar alle gebouwtypes
df_Mi_Utility = pd.read_excel(data,'Utility')

#Residential subtype material intensity data points
df_Mi_Single = pd.read_excel(data,'Single')
df_Mi_Row = pd.read_excel(data,'Row')
df_Mi_Apartment = pd.read_excel(data,'Apartment')
df_Mi_High_rise = pd.read_excel(data,'High_rise')

#Office subtype material intensity data points
df_Mi_Office = pd.read_excel(data,'Office')
df_Mi_Commercial = pd.read_excel(data,'Commercial')
df_Mi_Other = pd.read_excel(data,'Other')

#%% Creating new columns in the dataframe

#Determine the length of other columns
sLength = len(df['Bouwjaar'])

#Copy usefull columns to new dataframe:
df_buildings = df[['Gebruiksfu', 'Bouwjaar', 'Pandhoogte','Shape_Le_1','Shape_Ar_1','geometry','Woningtype']].copy()
#df_buildings['Shape_Ar_1'] = df_buildings['Shape_Area']

#Take column length and use it to create new columns
sLength = len(df_buildings['Bouwjaar'])

#Add columns of identical length
df_buildings['Btype'] = np.zeros(sLength)
df_buildings['Subtype'] = np.zeros(sLength)
df_buildings['Floor_area'] = np.zeros(sLength)

#%% filters to extract outliers and calculate functional floor area
df_buildings = df_buildings[df_buildings['Shape_Ar_1'] <100000]
df_buildings = df_buildings[df_buildings['Shape_Ar_1'] >=10]
df_buildings = df_buildings[df_buildings['Floor_area'] >=0]
sLength2 = len(df_buildings['Bouwjaar'])

#Calculate total floor area for the building, assumption is 2.7 m per floor
df_buildings['Floor_area'] = (df_buildings['Shape_Ar_1']*df_buildings['Pandhoogte'])/2.7

#%% Make building type categories

#Residential & subtypes, subtypes zijn gehaald uit de woningtype dataset van de overheid
df_buildings.loc[df_buildings.Woningtype == 'vrijstaande woning', 'Subtype'] = 'Single'
df_buildings.loc[df_buildings.Woningtype == 'twee-onder-een-kap', 'Subtype'] = 'Single'
```

```
df_buildings.loc[df_buildings.Woningtype == 'hoekwoning', 'Subtype'] = 'Single'
df_buildings.loc[df_buildings.Woningtype == 'tussenwoning/geschakeld', 'Subtype'] = 'Row'
df_buildings.loc[df_buildings.Woningtype == 'appartement', 'Subtype'] = 'apartment'
df_buildings.loc[(df_buildings.Gebruiksfu == 'woonfunctie') & (df_buildings.Pandhoogte > 20 ), 'Subtype'] = 'High_rise'

#Utility & subtypes, subtypes zijn gehaald uit de BAG3D dataset
df_buildings.loc[df_buildings.Gebruiksfu != 'woonfunctie', 'Btype'] = 'Utility'
df_buildings.loc[df_buildings.Gebruiksfu == 'kantoorfunctie', 'Subtype'] = 'Office'
df_buildings.loc[df_buildings.Gebruiksfu == 'winkelfunctie', 'Subtype'] = 'Commercial'
df_buildings.loc[(df_buildings.Gebruiksfu != 'kantoorfunctie') & (df_buildings.Gebruiksfu != 'winkelfunctie') & (df_buildings.Btype == 'Utility'), 'Subtype'] = 'Other'

#%% Material columns

#Import all the materials mentioned in the excel sheet used to make df_Mi, and apply it to make the amount and names of the new columns.
for col in df_Mi_Residential.columns:

df_buildings[col] = np.zeros(sLength2)

#%% Define the material intensities function

column_name = df_Mi_Residential.columns [0] #hier worden de kolom namen (materialen) gekopieerd om te gebruiken voor identificatie in de functie

def calcMaterial():

for i in df_buildings.index: #i is de for loop door de kolommen van de df_buildings index

valt = df_buildings.at[i,'Subtype'] #val for building type

val = df_buildings.at[i,'Bouwjaar'] #val for boujaar

if val <= 1945:

if valt == 'Single':

for j in range(len(df_Mi_Single.columns)): #j is de for loop door de kolommen van df_Mi

column_name = df_Mi_Single.columns[j] #Hier wordt de huidige positie naam van j gepakt

df_buildings.at[i, column_name] = df_buildings.at[i,'Floor_area'] * df_Mi_Single.at[0 ,column_name]

if valt == 'Row':

for j in range(len(df_Mi_Row.columns)): #j is de for loop door de kolommen van df_Mi

column_name = df_Mi_Row.columns[j] #Hier wordt de huidige positie naam van j gepakt

df_buildings.at[i, column_name] = df_buildings.at[i,'Floor_area'] * df_Mi_Row.at[0 ,column_name]

if valt == 'Apartment':
```

```
for j in range(len(df_Mi_Apartment.columns)): #j is de for loop door de kolommen van df_Mi

column_name = df_Mi_Apartment.columns[j] #Hier wordt de huidige positie naam van j gepakt

df_buildings.at[i, column_name] = df_buildings.at[i,'Floor_area'] * df_Mi_Apartment.at[0,column_name]

if valt == 'High_rise':

for j in range(len(df_Mi_High_rise.columns)): #j is de for loop door de kolommen van df_Mi

column_name = df_Mi_High_rise.columns[j] #Hier wordt de huidige positie naam van j gepakt

df_buildings.at[i, column_name] = df_buildings.at[i,'Floor_area'] * df_Mi_High_rise.at[0,column_name]

if valt == 'Office':

for j in range(len(df_Mi_Office.columns)): #13 materials

column_name = df_Mi_Office.columns[j]

df_buildings.at[i, column_name] = df_buildings.at[i,'Floor_area'] * df_Mi_Office.at[0,column_name]

if valt == 'Commercial':

for j in range(len(df_Mi_Commercial.columns)): #13 materials

column_name = df_Mi_Commercial.columns[j]

df_buildings.at[i, column_name] = df_buildings.at[i,'Floor_area'] * df_Mi_Commercial.at[0,column_name]

if valt == 'Other':

for j in range(len(df_Mi_Other.columns)): #13 materials

column_name = df_Mi_Other.columns[j]

df_buildings.at[i, column_name] = df_buildings.at[i,'Floor_area'] * df_Mi_Other.at[0,column_name]

if val <= 1975 and val >1945:

if valt == 'Single':

for j in range(len(df_Mi_Single.columns)): #j is de for loop door de kolommen van df_Mi

column_name = df_Mi_Single.columns[j] #Hier wordt de huidige positie naam van j gepakt

df_buildings.at[i, column_name] = df_buildings.at[i,'Floor_area'] * df_Mi_Single.at[1,column_name]

if valt == 'Row':

for j in range(len(df_Mi_Row.columns)): #j is de for loop door de kolommen van df_Mi

column_name = df_Mi_Row.columns[j] #Hier wordt de huidige positie naam van j gepakt
```

```
df_buildings.at[i, column_name] = df_buildings.at[i,'Floor_area'] * df_Mi_Row.at[1,column_name]

if valt == 'Apartment':

for j in range(len(df_Mi_Apartment.columns)): #j is de for loop door de kolommen van df_Mi

column_name = df_Mi_Apartment.columns[j] #Hier wordt de huidige positie naam van j gepakt

df_buildings.at[i, column_name] = df_buildings.at[i,'Floor_area'] * df_Mi_Apartment.at[1,column_name]

if valt == 'High_rise':

for j in range(len(df_Mi_High_rise.columns)): #j is de for loop door de kolommen van df_Mi

column_name = df_Mi_High_rise.columns[j] #Hier wordt de huidige positie naam van j gepakt

df_buildings.at[i, column_name] = df_buildings.at[i,'Floor_area'] * df_Mi_High_rise.at[1,column_name]

if valt == 'Office':

for j in range(len(df_Mi_Office.columns)): #13 materials

column_name = df_Mi_Office.columns[j]

df_buildings.at[i, column_name] = df_buildings.at[i,'Floor_area'] * df_Mi_Office.at[1,column_name]

if valt == 'Commercial':

for j in range(len(df_Mi_Commercial.columns)): #13 materials

column_name = df_Mi_Commercial.columns[j]

df_buildings.at[i, column_name] = df_buildings.at[i,'Floor_area'] * df_Mi_Commercial.at[1,column_name]

if valt == 'Other':

for j in range(len(df_Mi_Other.columns)): #13 materials

column_name = df_Mi_Other.columns[j]

df_buildings.at[i, column_name] = df_buildings.at[i,'Floor_area'] * df_Mi_Other.at[1,column_name]

if val <= 2000 and val >1975:

if valt == 'Single':

for j in range(len(df_Mi_Single.columns)): #j is de for loop door de kolommen van df_Mi

column_name = df_Mi_Single.columns[j] #Hier wordt de huidige positie naam van j gepakt

df_buildings.at[i, column_name] = df_buildings.at[i,'Floor_area'] * df_Mi_Single.at[2,column_name]
```

```
if valt == 'Row':

for j in range(len(df_Mi_Row.columns)): #j is de for loop door de kolommen van df_Mi

column_name = df_Mi_Row.columns[j] #Hier wordt de huidige positie naam van j gepakt

df_buildings.at[i, column_name] = df_buildings.at[i,'Floor_area'] * df_Mi_Row.at[2,column_name]

if valt == 'Apartment':

for j in range(len(df_Mi_Apartment.columns)): #j is de for loop door de kolommen van df_Mi

column_name = df_Mi_Apartment.columns[j] #Hier wordt de huidige positie naam van j gepakt

df_buildings.at[i, column_name] = df_buildings.at[i,'Floor_area'] * df_Mi_Apartment.at[2,column_name]

if valt == 'High_rise':

for j in range(len(df_Mi_High_rise.columns)): #j is de for loop door de kolommen van df_Mi

column_name = df_Mi_High_rise.columns[j] #Hier wordt de huidige positie naam van j gepakt

df_buildings.at[i, column_name] = df_buildings.at[i,'Floor_area'] * df_Mi_High_rise.at[2,column_name]

if valt == 'Office':

for j in range(len(df_Mi_Office.columns)): #13 materials

column_name = df_Mi_Office.columns[j]

df_buildings.at[i, column_name] = df_buildings.at[i,'Floor_area'] * df_Mi_Office.at[2,column_name]

if valt == 'Commercial':

for j in range(len(df_Mi_Commercial.columns)): #13 materials

column_name = df_Mi_Commercial.columns[j]

df_buildings.at[i, column_name] = df_buildings.at[i,'Floor_area'] * df_Mi_Commercial.at[2,column_name]

if valt == 'Other':

for j in range(len(df_Mi_Other.columns)): #13 materials

column_name = df_Mi_Other.columns[j]

df_buildings.at[i, column_name] = df_buildings.at[i,'Floor_area'] * df_Mi_Other.at[2,column_name]

if val >2000:

if valt == 'Single':

for j in range(len(df_Mi_Single.columns)): #j is de for loop door de kolommen van df_Mi
```

```
column_name = df_Mi_Single.columns[j] #Hier wordt de huidige positie naam van j gepakt

df_buildings.at[i, column_name] = df_buildings.at[i,'Floor_area'] * df_Mi_Single.at[3,column_name]

if valt == 'Row':

for j in range(len(df_Mi_Row.columns)): #j is de for loop door de kolommen van df_Mi

column_name = df_Mi_Row.columns[j] #Hier wordt de huidige positie naam van j gepakt

df_buildings.at[i, column_name] = df_buildings.at[i,'Floor_area'] * df_Mi_Row.at[3,column_name]

if valt == 'Apartment':

for j in range(len(df_Mi_Apartment.columns)): #j is de for loop door de kolommen van df_Mi

column_name = df_Mi_Apartment.columns[j] #Hier wordt de huidige positie naam van j gepakt

df_buildings.at[i, column_name] = df_buildings.at[i,'Floor_area'] * df_Mi_Apartment.at[3,column_name]

if valt == 'High_rise':

for j in range(len(df_Mi_High_rise.columns)): #j is de for loop door de kolommen van df_Mi

column_name = df_Mi_High_rise.columns[j] #Hier wordt de huidige positie naam van j gepakt

df_buildings.at[i, column_name] = df_buildings.at[i,'Floor_area'] * df_Mi_High_rise.at[3,column_name]

if valt == 'Office':

for j in range(len(df_Mi_Office.columns)): #13 materials

column_name = df_Mi_Office.columns[j]

df_buildings.at[i, column_name] = df_buildings.at[i,'Floor_area'] * df_Mi_Office.at[3,column_name]

if valt == 'Commercial':

for j in range(len(df_Mi_Commercial.columns)): #13 materials

column_name = df_Mi_Commercial.columns[j]

df_buildings.at[i, column_name] = df_buildings.at[i,'Floor_area'] * df_Mi_Commercial.at[3,column_name]

if valt == 'Other':

for j in range(len(df_Mi_Other.columns)): #13 materials

column_name = df_Mi_Other.columns[j]

df_buildings.at[i, column_name] = df_buildings.at[i,'Floor_area'] * df_Mi_Other.at[3,column_name]
```

```
calcMaterial()

for j in range(len(df_Mi_Single.columns)):

column_name = df_Mi_Single.columns[j]

print("Stock", df_Mi_Single.columns[j], (df_buildings[column_name].sum()), 'tonnes')


#%% Export to Shapefile
#Write dataframes back to shapefiles:
df_buildings.to_file(driver='ESRI Shapefile',filename=r'C:\Users\Teun\Documents\Spatial data\Leiden\Leiden_materials.shp')


#%% end of code
print('Done')
```