ČESKÉ VYSOKÉ
UČENÍ TECHNICKÉ
V PRAZE

FAKULTA STAVEBNÍ

HABILITAČNÍ PRÁCE

PŘÍLOHA 1: METODOLOGIE ZPRACOVÁNÍ DAT

2026

ING. VJAČESLAV USMANOV, PH.D.

OBSAH

Data Science 01: Příprava a čištění dat (Data Preparation / Cleaning)

Získávání dat (Data Acquisition)

In [1]:
```
# Instalace potřebných knihoven
#%pip install pandas
#%pip install numpy
```

In [2]:
```
# Import potřebných knihoven
import pandas as pd
import numpy as np
```

Dataset `data_timelaps`, vzniklý na základě časosběrného monitorování robotických zdicích prací, je uložen ve formátu CSV. Obsahuje dobu pracovního cyklu bez stochastických prodlev.

In [3]:
```
# Soubor je načten a přiřazen do proměnné ‚df_timelaps‘
other_path = '../../data/00_Raw/data_timelaps.csv'
df_timelaps = pd.read_csv(other_path, header=0)
```

In [4]:
```
# Zobrazení prvních 5 řádků datasetu
print('Prvních 5 řádků datového rámce')
df_timelaps.head(5)
```

```
Prvních 5 řádků datového rámce
```

Out[4]:

	id	time
0	1	44
1	2	35
2	4	36
3	5	34
4	7	34

Dataset `data_ARSS`, vzniklý na základě výstupu z SW ARSS pro robotické zdicí práce, je uložen ve formátu CSV. Obsahuje digitální kladečský plán vytvořený ve SW ARSS.

In [5]:
```
# Soubor je načten a přiřazen do proměnné ‚df_ARSS‘
other_path = '../../data/00_Raw/data_ARSS.csv'
df_ARSS = pd.read_csv(other_path, header=0)
```

In [6]:
```
# Zobrazení prvních 5 řádků datasetu
print('Prvních 5 řádků datového rámce')
df_ARSS.head(5)
```

```
Prvních 5 řádků datového rámce
```

Out[6]:

	ID	TYPE	ROTATION	X	Y	Z	LAYER	PALLET
0	1	2	90	220.0	95.0	0	1	1
1	2	3	90	220.0	252.5	0	1	1
2	3	1	90	220.0	440.0	0	1	1
3	4	1	90	220.0	690.0	0	1	2
4	5	1	90	220.0	940.0	0	1	3

Dataset `data_delay`, vzniklý na základě časosběrného monitorování robotických zdicích prací, je uložen ve formátu CSV. Obsahuje stochastické prodlevy: defekt materiálu, servis, porucha systému, nutnost otáčení zdicích prvků.

In [7]:
```
# Soubor je načten a přiřazen do proměnné ‚df_delay‘
other_path = '../../data/00_Raw/data_delay.csv'
df_delay = pd.read_csv(other_path, header=0)
```

In [8]:
```
# Zobrazení prvních 5 řádků datasetu
print('Prvních 5 řádků datového rámce')
df_delay.head(5)
```

```
Prvních 5 řádků datového rámce
```

Out[8]:

	id	time	delay	total_time	type_delay
0	3	36	109	145	material
1	6	34	62	96	service
2	20	40	200	240	material
3	31	52	198	250	system
4	35	42	24	66	rotation

Přidání nebo změna názvů sloupců

In [9]:
```
# Tvorba názvů sloupců
headers = ['id', 'x', 'y', 'z']
print('Sloupce\n', headers)
```

```
Sloupce
 ['id', 'x', 'y', 'z']
```

In [10]:
```
# Nahrazení názvů sloupců a následná kontrola datového rámce
df_ARSS = df_ARSS[['ID', 'X', 'Y', 'Z']]
df_ARSS.columns = headers
df_ARSS.head()
```

Out[10]:

	id	x	y	z
0	1	220.0	95.0	0
1	2	220.0	252.5	0
2	3	220.0	440.0	0
3	4	220.0	690.0	0
4	5	220.0	940.0	0

Sloučení datasetů dle `id`

In [11]:
```
df_merged = df_ARSS.merge(df_timelaps, on="id", how="left")
df_merged = df_merged.merge(df_delay, on="id", how="left")
df_merged = df_merged.fillna(0)
df_merged['time'] = df_merged['time_x'] + df_merged['time_y']
df_merged['total_time'] = df_merged['time'] + df_merged['delay']
df_merged = df_merged.drop(["time_x", "time_y"], axis=1)
df_merged = df_merged[df_merged['total_time']>0]
df_merged = df_merged.reset_index(drop=True)
df = df_merged[['id', 'x', 'y', 'z', 'time', 'delay', 'type_delay', 'total_time']]
df
```

Out[11]:

	id	x	y	z	time	delay	type_delay	total_time
0	1	220.0	95.0	0	44.0	0.0	0	44.0
1	2	220.0	252.5	0	35.0	0.0	0	35.0
2	3	220.0	440.0	0	36.0	109.0	material	145.0
3	4	220.0	690.0	0	36.0	0.0	0	36.0
4	5	220.0	940.0	0	34.0	0.0	0	34.0
...	...	...	...	...	...	...	...	...
264	272	3440.0	220.0	2250	37.0	0.0	0	37.0
265	273	3690.0	220.0	2250	39.0	0.0	0	39.0
266	275	220.0	752.5	2250	41.0	41.0	rotation	82.0
267	276	220.0	3127.5	2250	44.0	0.0	0	44.0
268	277	220.0	3315.0	2250	26.0	0.0	0	26.0

269 rows × 8 columns

Analýza chybějících hodnot v datové sadě

In [12]:
```
# Logická hodnota ‚True‘ indikuje přítomnost chybějící hodnoty, zatímco ‚False‘ označuje její
missing_data = df.isnull()
missing_data.head(5)
```

Out[12]:

	id	x	y	z	time	delay	type_delay	total_time
0	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False

In [13]:
```
# Výpočet počtu chybějících hodnot v jednotlivých sloupcích datového rámce
for column in missing_data.columns.values.tolist():
    print(f'{missing_data[column].value_counts()}\n')
```

```
id
False    269
Name: count, dtype: int64

x
False    269
Name: count, dtype: int64

y
False    269
Name: count, dtype: int64

z
False    269
Name: count, dtype: int64

time
False    269
Name: count, dtype: int64

delay
False    269
Name: count, dtype: int64

type_delay
False    269
Name: count, dtype: int64

total_time
False    269
Name: count, dtype: int64
```

Práce s chybějícími daty

Jak pracovat s chybějícími daty?

1. Odstranění dat:
   a. Odstranění celého řádku
   b. Odstranění celého sloupce
2. Nahrazení dat:
   a. Nahrazení průměrnou hodnotou
   b. Nahrazení nejčastější hodnotou (frekvencí)
   c. Nahrazení na základě jiných funkcí

Výpočty a následné úpravy dat

1. Change "type_delay" data to category values:
   a. material -> 1
   b. service -> 2
   c. rotation -> 3
   d. system -> 4

In [14]:
```
# úprava formátu dat: type_delay -> Int
df['type_delay'] = df['type_delay'].map({
    0: 0,
    'material': 1,
    'service': 2,
```

```
    'rotation': 3,
    'system': 4
})
df
```

Out[14]:

	id	x	y	z	time	delay	type_delay	total_time
0	1	220.0	95.0	0	44.0	0.0	0	44.0
1	2	220.0	252.5	0	35.0	0.0	0	35.0
2	3	220.0	440.0	0	36.0	109.0	1	145.0
3	4	220.0	690.0	0	36.0	0.0	0	36.0
4	5	220.0	940.0	0	34.0	0.0	0	34.0
...	...	...	...	...	...	...	...	...
264	272	3440.0	220.0	2250	37.0	0.0	0	37.0
265	273	3690.0	220.0	2250	39.0	0.0	0	39.0
266	275	220.0	752.5	2250	41.0	41.0	3	82.0
267	276	220.0	3127.5	2250	44.0	0.0	0	44.0
268	277	220.0	3315.0	2250	26.0	0.0	0	26.0

269 rows × 8 columns

Kontrola a úprava formátu dat

In [15]:
```
# Kontrola datového typu
df.dtypes
```

Out[15]:
```
id             int64
x            float64
y            float64
z              int64
time         float64
delay        float64
type_delay     int64
total_time   float64
dtype: object
```

In [16]:
```
df = df.astype(int)
```

In [17]:
```
# Kontrola datového typu
df.dtypes
```

Out[17]:
```
id           int64
x            int64
y            int64
z            int64
time         int64
delay        int64
type_delay   int64
total_time   int64
dtype: object
```

In [18]:
```
df.head()
```

Out[18]:

	id	x	y	z	time	delay	type_delay	total_time
0	1	220	95	0	44	0	0	44
1	2	220	252	0	35	0	0	35
2	3	220	440	0	36	109	1	145
3	4	220	690	0	36	0	0	36
4	5	220	940	0	34	0	0	34

In [19]:
```
df.describe()
```

Out[19]:

	id	x	y	z	time	delay	type_delay	total_ti
count	269.000000	269.000000	269.000000	269.000000	269.000000	269.000000	269.000000	269.000
mean	139.881041	1297.936803	1052.033457	1001.858736	36.360595	5.553903	0.245353	41.914
std	79.394072	1267.084840	1325.817766	707.763639	6.630443	26.319540	0.800668	27.564
min	1.000000	95.000000	95.000000	0.000000	22.000000	0.000000	0.000000	22.000
25%	72.000000	220.000000	220.000000	500.000000	32.000000	0.000000	0.000000	32.000
50%	139.000000	690.000000	220.000000	1000.000000	36.000000	0.000000	0.000000	36.000
75%	209.000000	2315.000000	1565.000000	1500.000000	40.000000	0.000000	0.000000	42.000
max	277.000000	4002.000000	4690.000000	2250.000000	58.000000	260.000000	4.000000	296.000

Identifikace odlehlých hodnot

In [20]:
```
# Nastavení horní a dolní meze pro odlehlé hodnoty
low_limit = 0.05
hi_limit =  0.95

q_low = df['time'].quantile(low_limit)
q_hi  = df['time'].quantile(hi_limit)

print(f'{q_low=}, {q_hi=}')
df_out_limit = df[(df['time'] > q_hi) | (df['time'] < q_low)]
df_out_limit
```

```
q_low=np.float64(26.0), q_hi=np.float64(48.599999999999994)
```

Out[20]:

	id	x	y	z	time	delay	type_delay	total_time
26	31	3065	220	0	52	198	4	250
27	32	3315	220	0	58	0	0	58
28	33	3565	220	0	53	0	0	53
38	43	1690	220	250	49	0	0	49
43	48	2940	220	250	53	0	0	53
45	50	3440	220	250	50	0	0	50
46	51	3690	220	250	50	0	0	50
65	70	220	95	500	55	0	0	55
85	90	815	220	500	56	0	0	56
100	105	252	220	750	53	0	0	53
111	116	2940	220	750	52	0	0	52
112	117	3190	220	750	55	0	0	55
135	140	220	690	1000	24	0	0	24
136	141	220	940	1000	24	24	3	48
137	142	220	3190	1000	24	0	0	24
138	143	220	3440	1000	22	0	0	22
140	145	220	3940	1000	24	0	0	24
141	146	220	4190	1000	24	0	0	24
200	208	3565	220	1500	49	0	0	49
223	231	220	3127	1750	24	0	0	24
228	236	220	252	2000	54	0	0	54

Výpočet nejistoty

In [21]:

```python
def measurement_uncertainty(df, column, delta_t=1, k=2):
    """
    Výpočet nejistoty měření z časosběrných dat.

    Parametry:
    df       : pandas DataFrame s daty
    column   : název sloupce s měřením (např. čas v s)
    delta_t  : časové rozlišení přístroje (s)
    k        : koeficient rozšíření (default k=2)

    Návratová hodnota:
    dict s výsledky
    """

    data = df[column].dropna()

    n = len(data)
    mean = data.mean()
    s = data.std(ddof=1)
```

```
    # Typ A
    u_A = s / np.sqrt(n)

    # Typ B (kvantizace)
    u_B_single = (delta_t / 2) / np.sqrt(3)
    u_B_mean = u_B_single / np.sqrt(n)

    # Kombinovaná
    u_c = np.sqrt(u_A**2 + u_B_mean**2)

    # Rozšířená
    U = k * u_c

    return {
        "serie": column,
        "n": n,
        "mean": mean,
        "std_dev": s,
        "u_A": u_A,
        "u_B_single": u_B_single,
        "u_B_mean": u_B_mean,
        "u_c": u_c,
        "U": U
    }

def report(result):
    print(f"{result['serie']}:")
    print(f"Kombinovaná nejistota: ({result['mean']:.2f} ± {result['u_c']:.2f})")
    print(f"Rozšířená nejistota: ({result['mean']:.2f} ± {result['U']:.2f}) s (k=2, 95%)")
    print(f"Nejistota Typ A (stochastická): {result['u_A']:.4f}")
    print(f"Nejistota Typ B (přístrojová / 1 měření): {result['u_B_single']:.4f}")
    print(f"Nejistota Typ B (přístrojová / průměr): {result['u_B_mean']:.4f}")
    print()
```

In [22]:
```
result = measurement_uncertainty(df, 'total_time', delta_t=1)
report(result)

result = measurement_uncertainty(df, 'time', delta_t=1)
report(result)
```

```
total_time:
Kombinovaná nejistota: (41.91 ± 1.68)
Rozšířená nejistota: (41.91 ± 3.36) s (k=2, 95%)
Nejistota Typ A (stochastická): 1.6806
Nejistota Typ B (přístrojová / 1 měření): 0.2887
Nejistota Typ B (přístrojová / průměr): 0.0176

time:
Kombinovaná nejistota: (36.36 ± 0.40)
Rozšířená nejistota: (36.36 ± 0.81) s (k=2, 95%)
Nejistota Typ A (stochastická): 0.4043
Nejistota Typ B (přístrojová / 1 měření): 0.2887
Nejistota Typ B (přístrojová / průměr): 0.0176
```

Export datové sady do formátu CSV

In [23]:
```
df.to_csv("../../data/01_DataScience/clean_timelaps.csv", index=False)
```

Read/Save Other Data Formats

Data Formate	Read	Save

Data Formate	Read	Save
csv	`pd.read_csv()`	`df.to_csv()`
json	`pd.read_json()`	`df.to_json()`
excel	`pd.read_excel()`	`df.to_excel()`
hdf	`pd.read_hdf()`	`df.to_hdf()`
sql	`pd.read_sql()`	`df.to_sql()`

Autor / Organizace / Datum

Vjačeslav Usmanov, ČVUT v Praze, Fakulta stavební

Přehled změn

Datum (YYYY-MM-DD)	Verze	Autor změny	Popis změny
2026-01-18	1.1	Vjačeslav Usmanov	added DS_01_Data_Cleaning.ipnyb
2026-02-10	1.2	Vjačeslav Usmanov	changed DS_01_Data_Cleaning.ipnyb

Data Science 02: Explorační analýza dat (Data Exploration)

In [1]:
```
# Instalace potřebných knihoven
#%pip install pandas
#%pip install numpy
#%pip install matplotlib
```

In [2]:
```
# Import potřebných knihoven
import pandas as pd
import numpy as np

import matplotlib as plt
from matplotlib import pyplot
```

In [3]:
```
# Soubor je načten a přiřazen do proměnné ‚df‘
other_path = "../../data/01_DataScience/clean_timelaps.csv"
df = pd.read_csv(other_path)
df.head()
```

Out[3]:

	id	x	y	z	time	delay	type_delay	total_time
0	1	220	95	0	44	0	0	44
1	2	220	252	0	35	0	0	35
2	3	220	440	0	36	109	1	145
3	4	220	690	0	36	0	0	36
4	5	220	940	0	34	0	0	34

Základní charakteristika datové sady

Datové typy

In [4]:
```
df.dtypes
```

Out[4]:
```
id             int64
x              int64
y              int64
z              int64
time           int64
delay          int64
type_delay     int64
total_time     int64
dtype: object
```

Popis datové sady

In [5]:
```
df.describe()
```

Out[5]:

	id	x	y	z	time	delay	type_delay	total_ti
count	269.000000	269.000000	269.000000	269.000000	269.000000	269.000000	269.000000	269.000
mean	139.881041	1297.936803	1052.033457	1001.858736	36.360595	5.553903	0.245353	41.914
std	79.394072	1267.084840	1325.817766	707.763639	6.630443	26.319540	0.800668	27.564
min	1.000000	95.000000	95.000000	0.000000	22.000000	0.000000	0.000000	22.000
25%	72.000000	220.000000	220.000000	500.000000	32.000000	0.000000	0.000000	32.000
50%	139.000000	690.000000	220.000000	1000.000000	36.000000	0.000000	0.000000	36.000
75%	209.000000	2315.000000	1565.000000	1500.000000	40.000000	0.000000	0.000000	42.000
max	277.000000	4002.000000	4690.000000	2250.000000	58.000000	260.000000	4.000000	296.000

Základní informace o datové sadě

In [6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 269 entries, 0 to 268
Data columns (total 8 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   id           269 non-null    int64
 1   x            269 non-null    int64
 2   y            269 non-null    int64
 3   z            269 non-null    int64
 4   time         269 non-null    int64
 5   delay        269 non-null    int64
 6   type_delay   269 non-null    int64
 7   total_time   269 non-null    int64
dtypes: int64(8)
memory usage: 16.9 KB
```

Proces standardizace dat (Data Standardization)

Proces normalizace dat (Data Normalization)

Normalizace představuje proces transformace hodnot vybraných proměnných do srovnatelného rozsahu. Typické přístupy zahrnují standardizaci na nulovou střední hodnotu, úpravu rozptylu na jednotkovou hodnotu nebo lineární škálování do intervalu ⟨0, 1⟩.

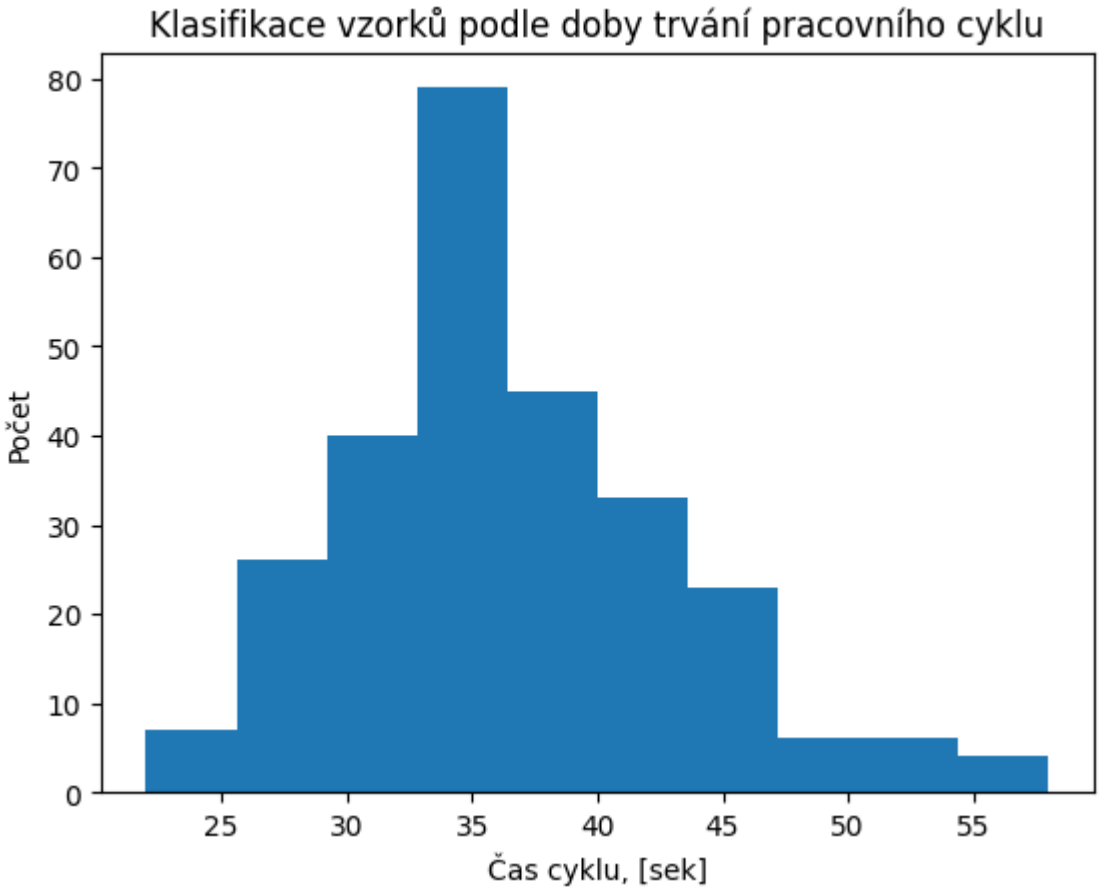
Diskretizace spojitých proměnných (binning)

In [7]:
```
%matplotlib inline

plt.pyplot.hist(df["time"])

plt.pyplot.xlabel("Čas cyklu, [sek]")
plt.pyplot.ylabel("Počet")
plt.pyplot.title("Klasifikace vzorků podle doby trvání pracovního cyklu")
```

Out[7]:
```
Text(0.5, 1.0, 'Klasifikace vzorků podle doby trvání pracovního cyklu')
```



In [8]:
```
# Definice intervalů (binů) pro diskrétní rozdělení dat
bins = np.linspace(min(df["time"]), max(df["time"]), 8)
bins
```

Out[8]:
```
array([22.        , 27.14285714, 32.28571429, 37.42857143, 42.57142857,
       47.71428571, 52.85714286, 58.        ])
```

In [9]:
```
# pojmenování intervalů (binů) pro diskrétní rozdělení dat
group_names = ['Extremely short','Very short','Short', 'Normal', 'Long','Very long','Extremely
```

In [10]:
```
# Kategorizace intervalů
df['time_binned'] = pd.cut(df['time'], bins, labels=group_names, include_lowest=True )
df[['time','time_binned']].head(20)
```

Out[10]:

	time	time_binned
0	44	Long
1	35	Short
2	36	Short
3	36	Short
4	34	Short
5	34	Short
6	34	Short
7	33	Short
8	36	Short
9	42	Normal
10	32	Very short
11	35	Short
12	32	Very short
13	47	Long
14	32	Very short
15	40	Normal
16	42	Normal
17	46	Long
18	48	Very long
19	44	Long

In [11]:
```
# Výpočet počtu vzorků v intervalech
df["time"].value_counts()
```

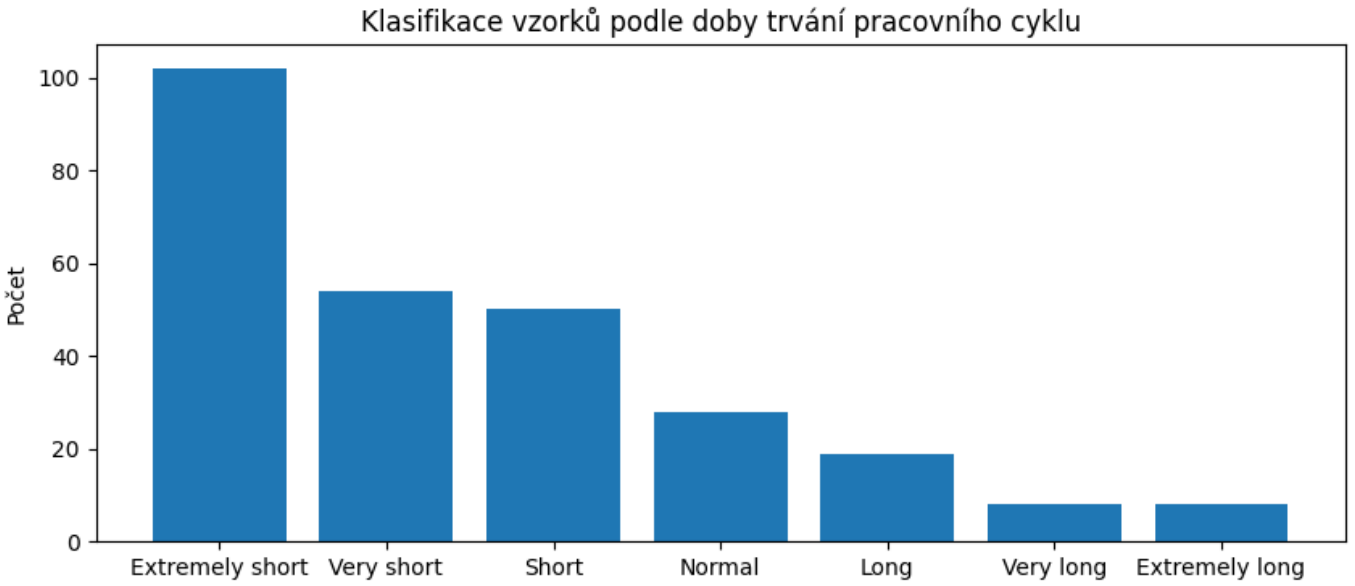
Out[11]:
```
time
34    23
37    23
33    21
36    18
35    17
32    16
38    13
31    12
30    12
42    10
41     9
46     9
28     9
39     9
40     9
26     8
45     6
24     6
44     5
29     5
43     5
27     4
47     3
53     3
48     2
55     2
52     2
49     2
50     2
58     1
56     1
22     1
54     1
Name: count, dtype: int64
```

Grafické znázornění intervalového rozdělení

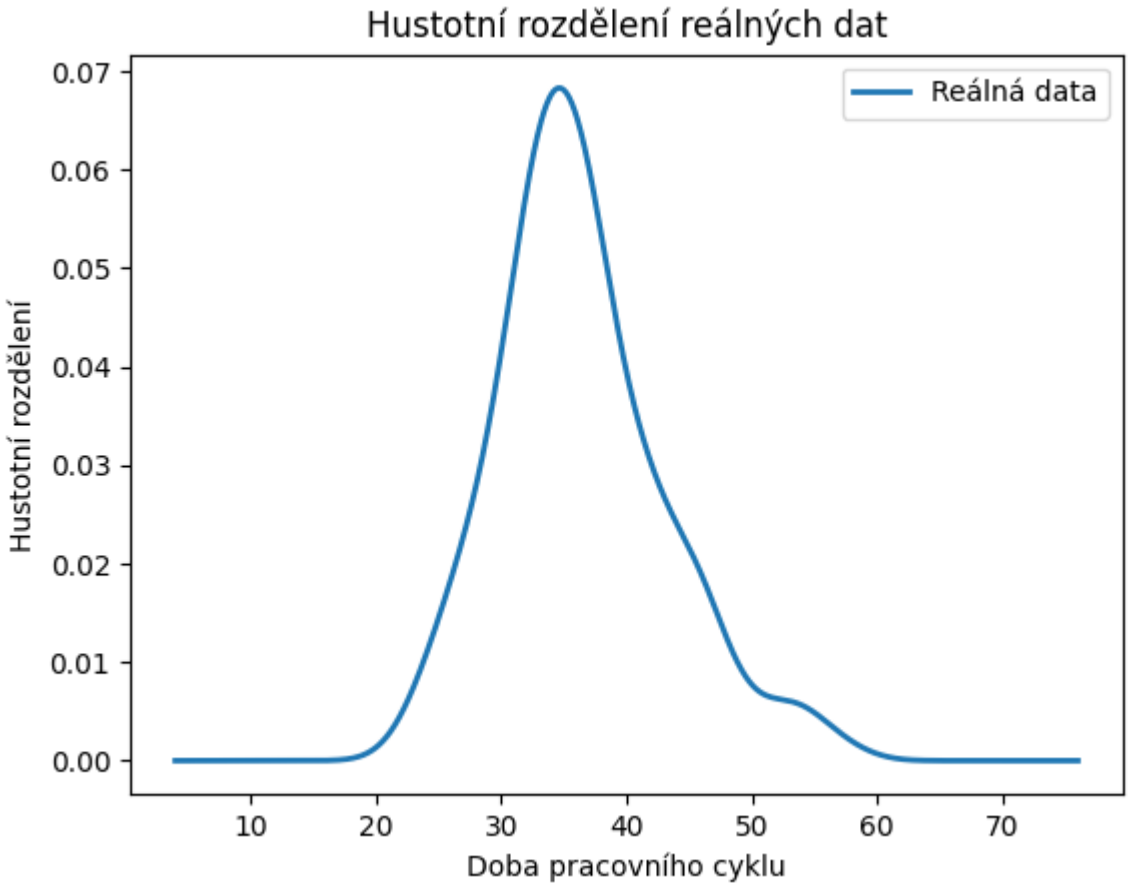
In [12]:
```
%matplotlib inline
pyplot.figure(figsize=(10,4))
pyplot.bar(group_names, df["time_binned"].value_counts(), )

pyplot.ylabel("Počet")
pyplot.title("Klasifikace vzorků podle doby trvání pracovního cyklu")
```

Out[12]:
```
Text(0.5, 1.0, 'Klasifikace vzorků podle doby trvání pracovního cyklu')
```



In [13]:
```
# Plot density
df['time'].plot.density(bw_method='scott', linestyle='-', linewidth=2, label='Reálná data')
pyplot.legend()
pyplot.xlabel("Doba pracovního cyklu")
pyplot.ylabel("Hustotní rozdělení")
pyplot.title('Hustotní rozdělení reálných dat')
pyplot.show()
```



Indikátorová (dummy) proměnná

In [14]:
```
df.columns
```

Out[14]:
```
Index(['id', 'x', 'y', 'z', 'time', 'delay', 'type_delay', 'total_time',
       'time_binned'],
      dtype='object')
```

In [15]:
```
dummy_variable_1 = pd.get_dummies(df["type_delay"])
```

```
dummy_variable_1.head()
```

Out[15]:

	0	1	2	3	4
0	True	False	False	False	False
1	True	False	False	False	False
2	False	True	False	False	False
3	True	False	False	False	False
4	True	False	False	False	False

In [16]:
```
# Změna názvů sloupců pro větší přehlednost
dummy_variable_1.rename(columns={0:'wo_delay', 1:'material', 2:'service', 3:'rotation', 4:'sys
dummy_variable_1.head()
```

Out[16]:

	wo_delay	material	service	rotation	system
0	True	False	False	False	False
1	True	False	False	False	False
2	False	True	False	False	False
3	True	False	False	False	False
4	True	False	False	False	False

In [17]:
```
# Sloučení datových rámců "df" a "dummy_variable_1"
df = pd.concat([df, dummy_variable_1], axis=1)
df.head()
```

Out[17]:

	id	x	y	z	time	delay	type_delay	total_time	time_binned	wo_delay	material	service	rot
0	1	220	95	0	44	0	0	44	Long	True	False	False	
1	2	220	252	0	35	0	0	35	Short	True	False	False	
2	3	220	440	0	36	109	1	145	Short	False	True	False	
3	4	220	690	0	36	0	0	36	Short	True	False	False	
4	5	220	940	0	34	0	0	34	Short	True	False	False	

Export datové sady do formátu CSV

In [18]:
```
df.to_csv('../../data/01_DataScience/exploration_timelaps.csv', index=False)
```

Autor / Organizace / Datum

Vjačeslav Usmanov, ČVUT v Praze, Fakulta stavební

Přehled změn

Datum (YYYY-MM-DD)	Verze	Autor změny	Popis změny
2026-01-20	1.1	Vjačeslav Usmanov	added DS_02_Exploration.ipynb
2026-02-11	1.2	Vjačeslav Usmanov	changed DS_02_Exploration.ipynb

Data Science 03: Inženýrství příznaků (Feature Engineering)

In [1]:
```
# Instalace potřebných knihoven
#%pip instal pandas
#%pip install numpy

#%pip install scipy

#%pip install seaborn
```

In [2]:
```
# Import potřebných knihoven
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
import seaborn as sns

from scipy import stats

import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)

%matplotlib inline
```

In [3]:
```
# Soubor je načten a přiřazen do proměnné ‚df‘
path='../../data/01_DataScience/exploration_timelaps.csv'
df = pd.read_csv(path)
df.head()
```

Out[3]:

	id	x	y	z	time	delay	type_delay	total_time	time_binned	wo_delay	material	service	rot
0	1	220	95	0	44	0	0	44	Long	True	False	False	
1	2	220	252	0	35	0	0	35	Short	True	False	False	
2	3	220	440	0	36	109	1	145	Short	False	True	False	
3	4	220	690	0	36	0	0	36	Short	True	False	False	
4	5	220	940	0	34	0	0	34	Short	True	False	False	

Analýza vzorců jednotlivých příznaků prostřednictvím grafické vizualizace

Výpočet korelace mezi proměnnými

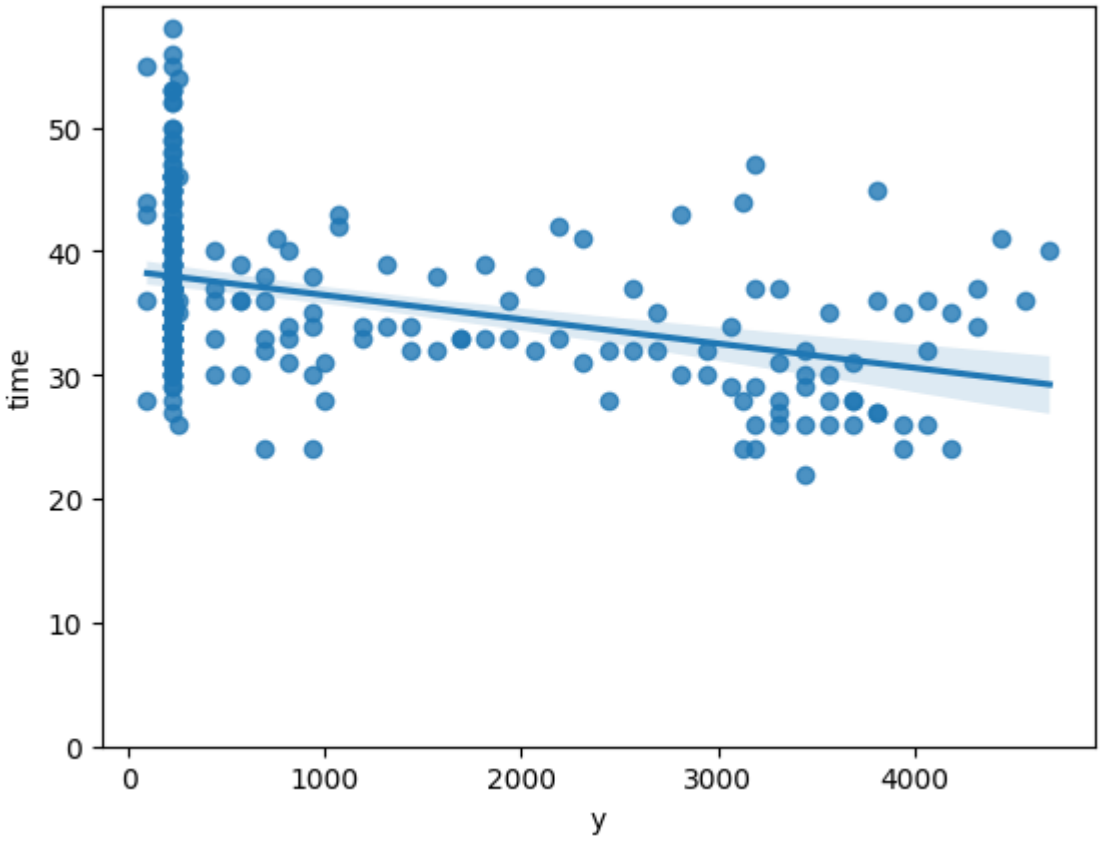
In [4]:
```
df[['x', 'y', 'z', 'time', 'delay', 'type_delay', 'total_time']].corr()
```

Out[4]:

	x	y	z	time	delay	type_delay	total_time
x	1.000000	-0.535873	0.120786	0.395179	0.025878	0.134066	0.119768
y	-0.535873	1.000000	-0.169543	-0.391292	-0.013002	-0.109445	-0.106539
z	0.120786	-0.169543	1.000000	-0.343239	-0.056292	0.023884	-0.136315
time	0.395179	-0.391292	-0.343239	1.000000	0.066204	0.035987	0.303759
delay	0.025878	-0.013002	-0.056292	0.066204	1.000000	0.591478	0.970769
type_delay	0.134066	-0.109445	0.023884	0.035987	0.591478	1.000000	0.573425
total_time	0.119768	-0.106539	-0.136315	0.303759	0.970769	0.573425	1.000000

In [5]:
```
# 'x' jako potenciální prediktor 'time'
sns.regplot(x="y", y="time", data=df)
plt.ylim(0,)
```

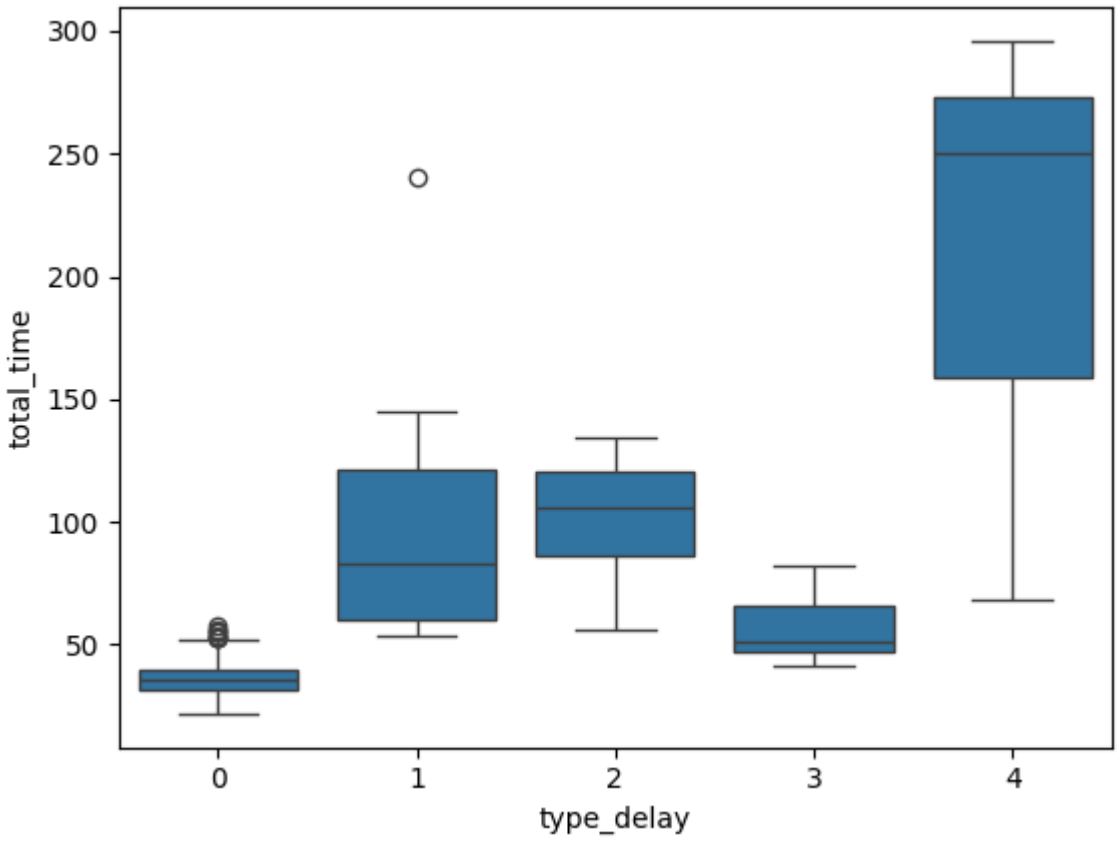
Out[5]: (0.0, 59.8)



Analýza kategorických proměnných

In [6]:
```
# Vztah mezi proměnnými ‚type_delay' a ‚total_time'
sns.boxplot(x='type_delay', y="total_time", data=df)
```

Out[6]: <Axes: xlabel='type_delay', ylabel='total_time'>



Deskriptivní statistická analýza dat

In [7]:
```
df.describe()
```

Out[7]:

	id	x	y	z	time	delay	type_delay	total_ti
count	269.000000	269.000000	269.000000	269.000000	269.000000	269.000000	269.000000	269.000
mean	139.881041	1297.936803	1052.033457	1001.858736	36.360595	5.553903	0.245353	41.914
std	79.394072	1267.084840	1325.817766	707.763639	6.630443	26.319540	0.800668	27.564
min	1.000000	95.000000	95.000000	0.000000	22.000000	0.000000	0.000000	22.000
25%	72.000000	220.000000	220.000000	500.000000	32.000000	0.000000	0.000000	32.000
50%	139.000000	690.000000	220.000000	1000.000000	36.000000	0.000000	0.000000	36.000
75%	209.000000	2315.000000	1565.000000	1500.000000	40.000000	0.000000	0.000000	42.000
max	277.000000	4002.000000	4690.000000	2250.000000	58.000000	260.000000	4.000000	296.000

Výpočet četnosti jednotlivých hodnot

In [8]:
```
df['type_delay'].value_counts()
```

Out[8]:
```
type_delay
0    242
3     13
1      7
2      4
4      3
Name: count, dtype: int64
```

In [9]:
```
df['type_delay'].value_counts().to_frame()
```

Out[9]:

type_delay	count
0	242
3	13
1	7
2	4
4	3

Seskupování dat

In [10]:
```
df['type_delay'].unique()
```

Out[10]:
```
array([0, 1, 2, 4, 3])
```

In [11]:
```
df_group_one = df[['type_delay', 'total_time']]
df_group_one
```

Out[11]:

	type_delay	total_time
0	0	44
1	0	35
2	1	145
3	0	36
4	0	34
...	...	...
264	0	37
265	0	39
266	3	82
267	0	44
268	0	26

269 rows × 2 columns

In [12]:
```
# Výpočet průměrné hodnoty času pro jednotlivé kategorie dat
df_group_one = df_group_one.groupby(['type_delay'], as_index=False).mean()
df_group_one
```

Out[12]:

	type_delay	total_time
0	0	36.359504
1	1	105.857143
2	2	100.500000
3	3	55.307692
4	4	204.666667

Vztah mezi korelací a kauzalitou

Korelace: míra vzájemné závislosti mezi proměnnými.

Kauzalita: vztah příčiny a následku mezi dvěma proměnnými.

Pearsonova korelace

Pearsonův korelační koeficient měří lineární závislost mezi dvěma proměnnými X a Y.

Výsledný koeficient nabývá hodnot v intervalu od -1 do 1, kde:

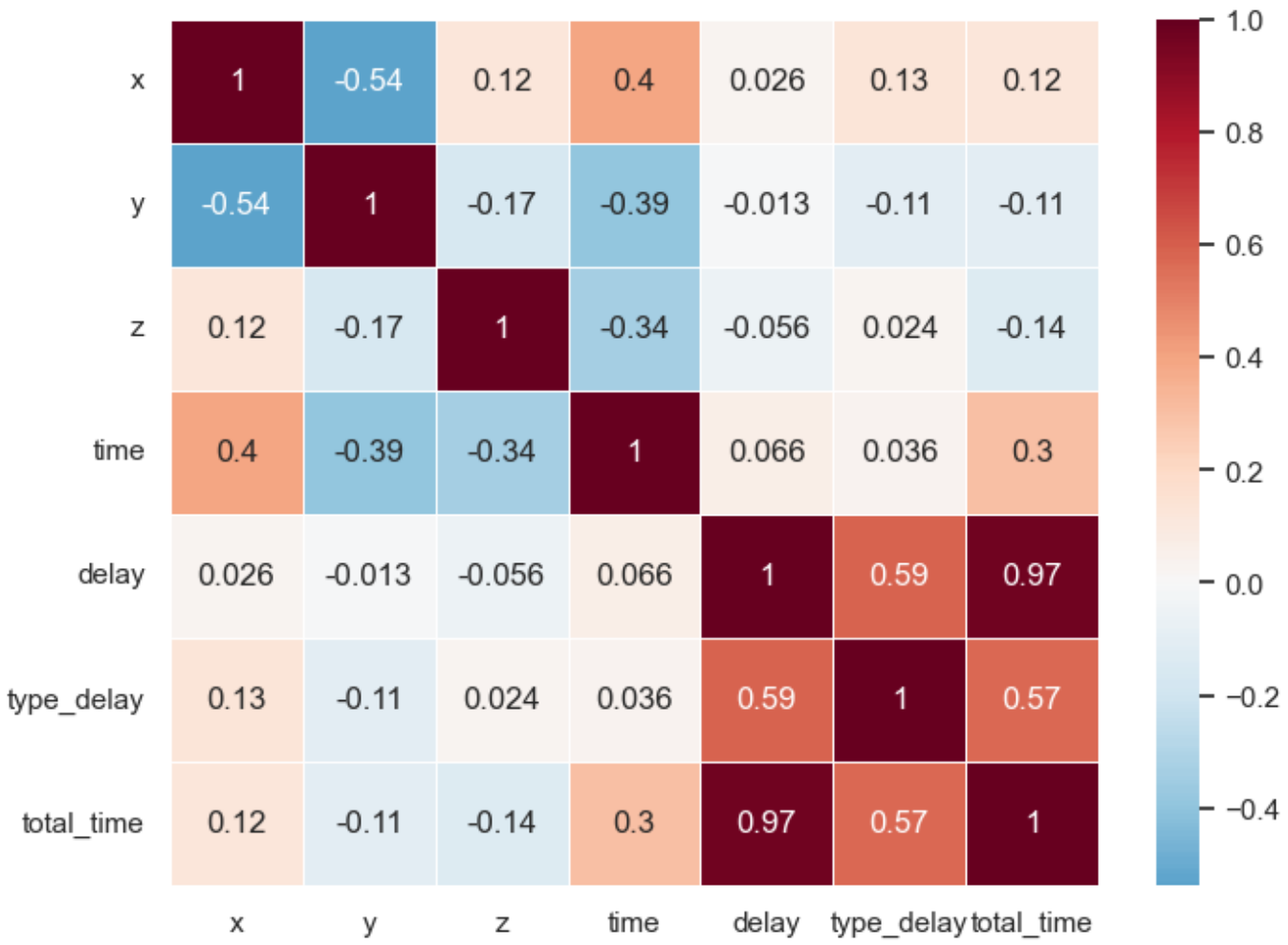
- **1**: Dokonalá kladná lineární korelace.
- **0**: Žádná lineární korelace, proměnné se pravděpodobně lineárně neovlivňují.
- **-1**: Dokonalá záporná lineární korelace.

In [13]:
```
corr = df[['x', 'y', 'z', 'time', 'delay', 'type_delay', 'total_time']].corr()
```

In [14]:
```
# Teplotní mapa (heatmapa)
sns.set_theme(style="white")

plt.figure(figsize=(8, 6))
sns.heatmap(
    corr,
    annot=True,
    cmap="RdBu_r",
    center=0,
    linewidths=0.5
)
```

Out[14]:
```
<Axes: >
```



P-value

P-hodnota (P-value) představuje pravděpodobnost, že korelace mezi dvěma proměnnými je statisticky významná. Obvykle se volí hladina významnosti 0,05, což znamená, že s 95% jistotou považujeme korelaci mezi proměnnými za statisticky významnou.

Podle běžně používané konvence platí, že pokud:

- p-hodnota je < 0,001: existuje silný důkaz, že korelace je statisticky významná.
- p-hodnota je < 0,05: existuje středně silný důkaz, že korelace je statisticky významná.
- p-hodnota je < 0,1: existuje slabý důkaz, že korelace je statisticky významná.
- p-hodnota je > 0,1: neexistuje důkaz o statistické významnosti korelace.

'dest_to_end' vs. 'total_time'

In [15]:
```
# Výpočet Pearsonova korelačního koeficientu a odpovídající p-hodnoty mezi proměnnými ‚x‘ a ‚t
pearson_coef, p_value = stats.pearsonr(df['x'], df['time'])
print("Pearsonov korelační koeficient = ", pearson_coef, ".  P-value =", p_value)
```

```
Pearsonov korelační koeficient =  0.3951785339509214 .  P-value = 1.7316685425319944e-11
```

Protože p-hodnota je < 0,001, korelace mezi proměnnými je statisticky významná a lineární vztah je středně silný

Analýza rozptylu (ANOVA) pro porovnání skupinových průměrů

Analýza rozptylu (ANOVA) je statistická metoda používaná k ověření, zda existují statisticky významné rozdíly mezi průměry dvou nebo více skupin. ANOVA vrací dva základní parametry:

F-statistika (F-test): ANOVA předpokládá, že průměry všech skupin jsou stejné, a následně vyhodnocuje,

jak moc se skutečné průměry od tohoto předpokladu odchylují. Tato odchylka je vyjádřena hodnotou F-statistiky. Vyšší hodnota znamená větší rozdíl mezi skupinovými průměry.

P-hodnota: P-hodnota udává, jak statisticky významná je vypočtená hodnota F-statistiky.

Pokud je analyzovaná proměnná silně korelována s vysvětlovanou proměnnou, očekáváme, že ANOVA vrátí vysokou hodnotu F-statistiky a nízkou p-hodnotu.

In [16]:
```
# provedeme seskupení dat podle jednotlivých kategorií
grouped_test = df[['type_delay', 'total_time']].groupby('type_delay')
grouped_test.head()
```

Out[16]:

	type_delay	total_time
0	0	44
1	0	35
2	1	145
3	0	36
4	0	34
5	2	96
6	0	34
15	1	240
26	4	250
30	3	66
47	3	69
72	2	56
73	1	63
98	3	60
99	4	68
115	3	67
136	3	48
152	1	83
160	1	58
169	2	134
179	4	296
219	2	116

In [17]:
```
grouped_test.get_group(1)['total_time']
```

Out[17]:
```
2      145
15     240
73      63
152     83
160     58
181     98
257     54
Name: total_time, dtype: int64
```

BASIC a HALF

In [18]:
```
f_val, p_val = stats.f_oneway(grouped_test.get_group(1)['total_time'], grouped_test.get_group(

print( "ANOVA results: F=", f_val, ", P =", p_val )
```

```
ANOVA results: F= 0.021705864420776055 , P = 0.8861199353688878
```

Výsledky analýzy ANOVA pro kategorie `material` a `service` vykazují p-hodnotu vyšší než 0,1, což znamená, že F-statistika není statisticky významná. Nelze tedy zamítnout nulovou hypotézu o shodě průměrů obou skupin a nelze potvrdit statisticky významný rozdíl mezi nimi.

Závěr: Identifikace významných proměnných

Na základě provedené analýzy byly identifikovány následující významné proměnné:

- total_time
- time
- x
- y
- z
- type_delay

Export datové sady do formátu CSV

In [19]:
```
df_ready = df[['id', 'type_delay', 'x', 'y', 'z', 'time', 'total_time']]
```

In [20]:
```
df_ready.to_csv('../../data/01_DataScience/ready_timelaps.csv', index=False)
```

Autor / Organizace / Datum

Vjačeslav Usmanov, ČVUT v Praze, Fakulta stavební

Přehled změn

Datum (YYYY-MM-DD)	Verze	Autor změny	Popis změny
2026-01-21	1.1	Vjačeslav Usmanov	added DS_03_Features.ipynb
2026-02-12	1.2	Vjačeslav Usmanov	changed DS_03_Features.ipynb

Data Science 04: Náhodné rozdělení dat na tréninkovou a validační množinu (Random Data Splitting)

Import libraries:

In [1]:
```
# Instalace potřebných knihoven
#%pip instal pandas
#%pip install numpy

#%pip install scipy
#%pip install seaborn

#%pip install scikit-learn
#%pip install matplotlib
#%pip install seaborn

# actual installed version of sklearn
#%pip show scikit-learn
```

In [2]:
```
# Import potřebných knihoven
import pandas as pd
import numpy as np

from sklearn.model_selection import train_test_split

import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)
```

In [3]:
```
# Soubor je načten a přiřazen do proměnné ‚df‘
path='../../data/01_DataScience/clean_timelaps.csv'
df = pd.read_csv(path)
df.head()
```

Out[3]:

	id	x	y	z	time	delay	type_delay	total_time
0	1	220	95	0	44	0	0	44
1	2	220	252	0	35	0	0	35
2	3	220	440	0	36	109	1	145
3	4	220	690	0	36	0	0	36
4	5	220	940	0	34	0	0	34

Náhodné rozdělení dat

In [4]:
```
# Nastavení náhodného semene (random seed) pro reprodukovatelné rozdělení dat
user_seed = 122

# Náhodné rozdělení dat na tréninkovou a validační množinu (60/40)
df_train, df_val = train_test_split(df, test_size=0.4, random_state=user_seed)
```

In [5]:
```
df_train.head()
```

Out[5]:

	id	x	y	z	time	delay	type_delay	total_time
145	150	1315	220	1000	29	0	0	29
70	75	220	1190	500	33	0	0	33
231	239	220	940	2000	35	6	3	41
191	199	1315	220	1500	36	0	0	36
46	51	3690	220	250	50	0	0	50

In [6]:
```
df_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 161 entries, 145 to 187
Data columns (total 8 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   id          161 non-null    int64
 1   x           161 non-null    int64
 2   y           161 non-null    int64
 3   z           161 non-null    int64
 4   time        161 non-null    int64
 5   delay       161 non-null    int64
 6   type_delay  161 non-null    int64
 7   total_time  161 non-null    int64
dtypes: int64(8)
memory usage: 11.3 KB
```

In [7]:
```
df_val.head()
```

Out[7]:

	id	x	y	z	time	delay	type_delay	total_time
12	13	220	2940	0	32	0	0	32
72	77	220	1690	500	33	23	2	56
212	220	2190	220	1750	35	0	0	35
100	105	252	220	750	53	0	0	53
40	45	2190	220	250	45	0	0	45

In [8]:
```
df_val.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 108 entries, 12 to 3
Data columns (total 8 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   id          108 non-null    int64
 1   x           108 non-null    int64
 2   y           108 non-null    int64
 3   z           108 non-null    int64
 4   time        108 non-null    int64
 5   delay       108 non-null    int64
 6   type_delay  108 non-null    int64
 7   total_time  108 non-null    int64
dtypes: int64(8)
memory usage: 7.6 KB
```

Export datové sady (train + valid) do formátu CSV

In [9]:
```
df_train.to_csv('../../data/01_DataScience/final_timelaps.csv', index=False)

df_train.to_csv('../../data/06_AI/train/train_timelaps.csv', index=False)
df_val.to_csv('../../data/06_AI/val/valid_timelaps.csv', index=False)
```

Autor / Organizace / Datum

Vjačeslav Usmanov, ČVUT v Praze, Fakulta stavební

Přehled změn

Datum (YYYY-MM-DD)	Verze	Autor změny	Popis změny
2026-01-21	1.1	Vjačeslav Usmanov	added DS_04_Data_Splitting.ipynb
2026-02-12	1.2	Vjačeslav Usmanov	changed DS_04_Data_Splitting.ipynb