EEE 111 1st AY 20-21

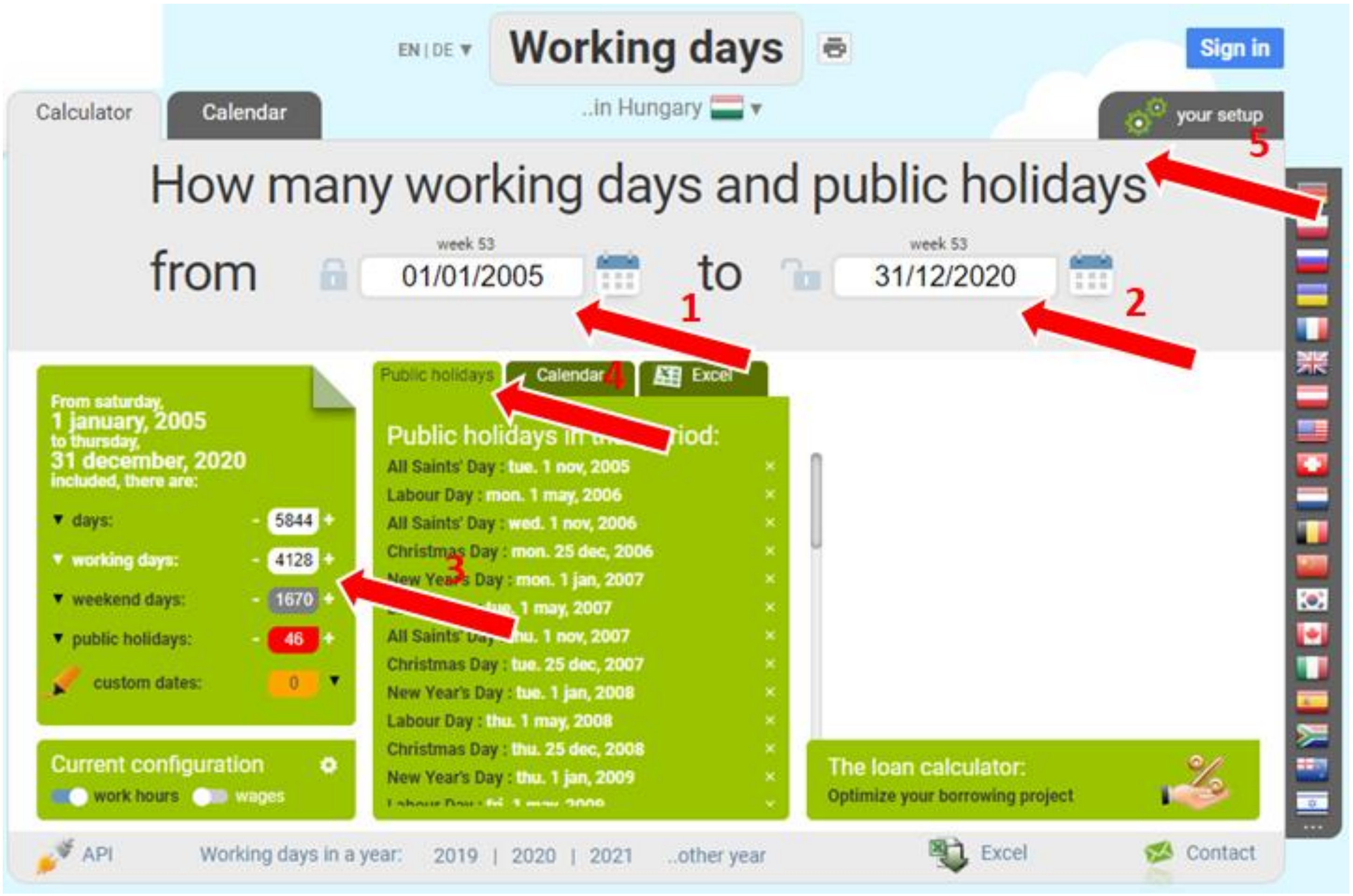
Machine Problem 1: Working Days Computer

- Please work **INDIVIDUALY**. Your codes will be run through a software similarity test and a plagiarism checker. Students who submitted substantially similar work or copied their codes online will be investigated for cheating will be filed a case with the College Disciplinary Council.
- Properly **INDENT** and **LABEL** your codes. Points will be deducted on codes with no proper indentation and labels.
- Save your codes in the following format: **MP1_LASTNAME.py** (e.g. MP1_ALPANO.py)
- Codes that does **NOT** compile will be given a grade of 0.
- Submit your .py files or .ipynb on or before 11:55 pm of **Nov 6, 2020 (Friday).** Codes not uploaded through Google Classroom and submitted through email will receive a deduction of 10% per week.

A **regular working day** is defined as any days except any Saturday, any Sunday, or any day which is declared as a holiday. Typically, it is described as a weekday without the holidays.

For this MP, you are required to compute the number of working days for a given time frame using the Gregorian Calendar. (https://en.wikipedia.org/wiki/Gregorian_calendar)

To see the correct number of working days, we can manually count using a calendar or reference the answer from the given website: (http://hungary.workingdays.org/)



How many working days and public holidays

from 01/01/2005 (week 53) to 31/12/2020 (week 53)

From saturday, 1 january, 2005 to thursday, 31 december, 2020 included, there are:

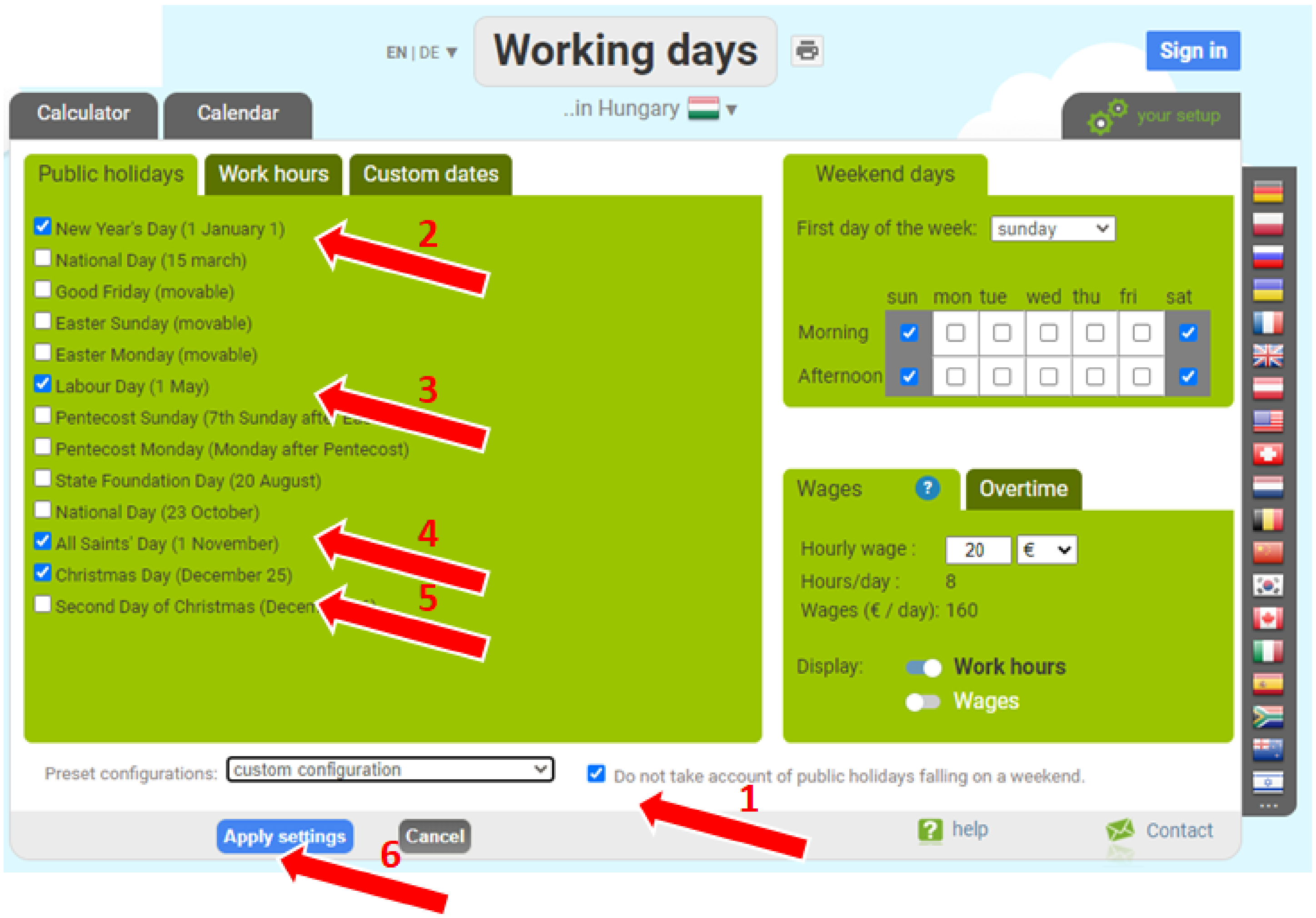
days:	5844
working days:	4128
weekend days:	1670
public holidays:	46
custom dates:	0

Public holidays in the period:

All Saints' Day : tue. 1 nov, 2005
Labour Day : mon. 1 may, 2006
All Saints' Day : wed. 1 nov, 2006
Christmas Day : mon. 25 dec, 2006
New Year's Day : mon. 1 jan, 2007
tue. 1 may, 2007
All Saints' Day : thu. 1 nov, 2007
Christmas Day : tue. 25 dec, 2007
New Year's Day : tue. 1 jan, 2008
Labour Day : thu. 1 may, 2008
Christmas Day : thu. 25 dec, 2008
New Year's Day : thu. 1 jan, 2009

Figure 1: hungary.workingdays.org Website Layout

Figure 1 shows the layout of http://hungary.workingdays.org/. It gives us the option to input the start day (Arrow 1) and end day (Arrow 2). Take note that it could only accept up to a day difference of 6000. After inputting the desired dates, a summary of the days between the start day and end day are displayed at the left window (Arrow 3) including the total days, total working days, total weekends, and total holidays. The holidays encountered during the given period are indicated at the first tab of the second window (Arrow 4). To set the wanted holidays, we can edit it on the setup tab (Arrow 5).



- [✓] New Year's Day (1 January 1)
- [ ] National Day (15 march)
- [ ] Good Friday (movable)
- [ ] Easter Sunday (movable)
- [ ] Easter Monday (movable)
- [✓] Labour Day (1 May)
- [ ] Pentecost Sunday (7th Sunday afte Ea
- [ ] Pentecost Monday (Monday after Pentecost)
- [ ] State Foundation Day (20 August)
- [ ] National Day (23 October)
- [✓] All Saints' Day (1 November)
- [✓] Christmas Day (December 25)
- [ ] Second Day of Christmas (Decen

First day of the week: sunday

	sun	mon	tue	wed	thu	fri	sat
Morning	✓	☐	☐	☐	☐	☐	✓
Afternoon	✓	☐	☐	☐	☐	☐	✓

Hourly wage : 20 €
Hours/day : 8
Wages (€ / day): 160
Display: [✓ Work hours] Wages

Preset configurations: custom configuration

[✓] Do not take account of public holidays falling on a weekend.

Figure 2: Setup Tab

We can set the desired holidays in the setup tab shown in figure 2. First, we need to tick the box that does not account holidays on weekends (Arrow 1). Next, we select the desired holidays (Arrow 2 to Arrow 5). For this Machine Problem, we have four holidays to consider: New Year (Jan 1), Labor Day (May 1), All Saint's Day (Nov 1), and Christmas Day (Dec 25). Lastly, do not forget to apply these settings (Arrow 6).

These are the information you need to take account:

1. In a single year, there are **365 days,** or roughly **52 weeks,** or **12 months** with each month having different number of days**.**
2. For a leap year, there is an extra day on February 29, having **366 days** total for the given year. "Every year that is exactly divisible by four is a leap year, except for years that are exactly divisible by 100, but these centurial years are leap years if they are exactly divisible by 400. For example, the years 1700, 1800, and 1900 are not leap years, but the years 1600 and 2000 are." (https://en.wikipedia.org/wiki/Gregorian_calendar)
3. Months with 31 days:
    a. January
    b. March
    c. May
    d. July
    e. August
    f. October
    g. December

   Months with 30 days:
    a. April
    b. June
    c. September
    d. November

   Months with 28 days (29 if leap year):
    a. February
4. There are only **5 working days for each week**, omitting Saturday and Sunday
5. Observed **holidays** which should **not be declared as a working day on a weekday** are:
    a. New year – every January 1 of each year
    b. Labor day – every May 1 of each year
    c. All Saints' day – every November 1 of each year
    d. Christmas – every December 25 of each year
6. We limit the **start date** to **January 1, 1971** which is a **Friday** and **end date** to **December 31, 2020** which is a **Thursday.**
7. **First leap year** occurs at the year **1972.**
8. The **reference website** (http://hungary.workingdays.org/) can only take into account **up to 6000 day difference**. Make sure your code can compute any 6000 day difference between January 1, 1971to December 31, 2020.

Specifications:

1. You are to **compute the total number of working days** between two given dates, defined as a weekday disregarding both holidays landing on a weekday and weekends (Saturday and Sunday).
2. You are allowed to use any standard libraries.
3. Your code should use the **"input()" command for each required date**. 1 for month, 1 for day, and 1 for year. (A total of 3 for start date and 3 for end date).
4. Your code should include the following functions with your own desired formal parameters:
    a. compute_total_days()
        i. Computes the total number of days from the start date to end date
        ii. Returns and/or prints the output of the given computation
    b. compute_weekdays()
        i. Computes the total number of weekdays from the start date to end date
        ii. Computes the total number of weekends from the start date to end date
        iii. Returns and/or prints the output of the given computations (both the number of weekdays and weekends)
    c. compute_leap_years()
        i. Computes the number of extra days from leap years between the start date to end date

ii. Take note that if the start date is after Feb 29 or the end date is before Feb 29, the additional day from the leap year should be ignored and should **NOT be considered** in the number of extra days from the leap year computation
iii. Returns and/or prints the output of the given computation

d. compute_holidays()

i. Computes the total number of holidays landing on a weekday
ii. Computes the number of New Year days between the start date to end date
iii. Computes the number of Labor days between the start date to end date
iv. Computes the number of All Saints' days between the start date to end date
v. Computes the number of Christmas days between the start date to end date
vi. Take note that **holidays landing on a weekend are NOT considered**
vii. Returns and/or prints the output of the given computations (number of New year days, Labor days, All Saints' days, and Christmas days with their total days altogether)

e. compute_workdays()

i. Computes the total number of workdays, defined as weekdays disregarding both the weekends and holidays landing on a weekday.
ii. Returns and/or prints the output of the given computation

Inputs:

1) Day of start date (integer)
2) Month of start date (integer)
3) Year of start date (integer)
4) Day of end date - inclusive (integer)
5) Month of end date (integer)
6) Year of end date (integer)

Outputs:

1) Number of days between the start day to end day, inclusive of the end date. (integer)
2) Number of weekdays and weekends between the start day to end day, inclusive of the end date. (integer)
3) Number of extra days from leap years. (integer)
4) Total number of holidays, including the total number of each holiday (integer)
5) Number of working days between the start date and end date, inclusive of the end date. (integer)

Error checking:

1) The input dates should be within the date limit (January 1, 1971 to December 31, 2020)
   - Warn the user that he has an input outside bounds, then end the program (no need to compute the working days)
   - You may exit the program as soon as it detects the error (rather than waiting for all 6 inputs)

Example:
```
Enter start month: 1
Enter start day: 1
Enter start year: 1971
Enter end month: 12
Enter end day: 31
Enter end year: 3030

Invalid input. Exiting Program.
```

2) The input dates should be valid (January has 31 days, December has 31 days, etc.)
   - Warn the user that he has an input outside bounds, then end the program (no need to compute the working days)
   - Date inputs of February 29 should have a corresponding leap year. Treat it as an error if the given year is not a leap year
   - You may exit the program as soon as it detects the error (rather than waiting for all 6 inputs)

Example:
```
Enter start month: 1
Enter start day: 33
Enter start year: 1971
Enter end month: 12
Enter end day: 31
Enter end year: 2020

Invalid input. Exiting Program.
```

3) All inputs should be an integer
   - Warn the user if he has an input which is not an integer, then end the program (no need to compute the working days)
   - You may exit the program as soon as it detects the error (rather than waiting for all 6 inputs)

Example:
```
Enter start month: a
Enter start day: 1
Enter start year: 1971
Enter end month: 12
Enter end day: 31
Enter end year: 2020

Invalid input. Exiting Program.
```

Milestones:

	Week 4	**Week 8**

Finish these functions at the end of the week	compute_total_days()
	compute_weekdays()
	compute_leap_years()
		compute_holidays()
		compute_workdays()

Grading Scheme:
+10% - Can compute and print total days between the given dates
+5% - Can compute and print total number of weekdays and weekends
+10% - Can compute and print additional days from leap year between inputs (Take note that if the start date is after Feb 29 or the end date is before Feb 29, the additional day from the leap year should be ignored)
+60% - Can compute total working days without holidays
15% - New Year
15% - Labor Day
15% - All Saints' Day
15% - Christmas Day
+15% - Can detect errors
5% - Within the date limit
5% - Valid input dates
5% - Integer inputs
100% - TOTAL

Examples:

#1

```
Enter start month: 4
Enter start day: 28
Enter start year: 1983
Enter end month: 2
Enter end day: 24
Enter end year: 1996

total days from start date to end date:  4686

total additional days from leap years:  3

total weekends:  1339

total days without weekends:  3347

new year holiday:  9
labor day holiday:  9
all saints day holiday:  10
christmas holiday:  9
total holidays:  37

total working days:  3310
```

#2

```
Enter start month: 12
Enter start day: 29
Enter start year: 2005
Enter end month: 6
Enter end day: 13
Enter end year: 2010

total days from start date to end date:  1628

total additional days from leap years:  1

total weekends:  466

total days without weekends:  1162

new year holiday:  4
labor day holiday:  4
all saints day holiday:  2
christmas holiday:  4
total holidays:  14

total working days:  1148
```

#3
```
Enter start month: 3
Enter start day: 19
Enter start year: 1971
Enter end month: 10
Enter end day: 3
Enter end year: 1986

total days from start date to end date:  5678

total additional days from leap years:  4

total weekends:  1622

total days without weekends:  4056

new year holiday:  10
labor day holiday:  11
all saints day holiday:  12
christmas holiday:  10
total holidays:  43

total working days:  4013
```

#4
```
Enter start month: 7
Enter start day: 7
Enter start year: 1977
Enter end month: 7
Enter end day: 7
Enter end year: 1978

total days from start date to end date:  366

total additional days from leap years:  0

total weekends:  104

total days without weekends:  262

new year holiday:  0
labor day holiday:  1
all saints day holiday:  1
christmas holiday:  0
total holidays:  2

total working days:  260
```