

Podstawy programowania

Zadania projektowe

Uwagi

- Zadania realizowane są w grupach.
- Wszystkie pliki i skrypty trafiają do systemu kontroli wersji. Obecnie najpopularniejszym wydaje się być **git** (<https://git-scm.com>). Podstawowe, darmowe rozwiązania można znaleźć pod adresami: <https://github.com>, <https://about.gitlab.com>, <https://bitbucket.org>.
- W przypadku opisu postępowania dla rozwiązania zadania (brak skryptu), opis powinien znajdować się w pliku **Markdown**. Zarówno **github** jak i **gitlab** obsługują dokumenty tego typu.
- Do zadań należy opracować sprawozdanie.
- Sprawozdanie należy wykonać zespołowo w *chmurze*. Popularne darmowe rozwiązania: <https://workspace.google.com/products/drive>, <https://www.dropbox.com/basic>, <https://www.microsoft.com/en-us/microsoft-365/onedrive/free-online-cloud-storage>.
- Finalne sprawozdanie należy oddać w formacie **PDF**, do sprawozdań dołączamy spakowany **klon repozytorium**.
- Linki do repozytoriów git i sprawozdań należy przysłać prowadzącemu.
- Zadania **od 3 do 9** można zrealizować (w różny sposób) za pomocą programów (poleceń) kolekcji **GNU** (<https://www.gnu.org/software>): **cat**, **grep**, **sed**, **awk**, **diff**, **patch**, **find**, **paste**, **md5sum** i dodatkowo: **magick**.
- Dwa ostatnie zadania będą wymagały napisania prostego skryptu iterującego po plikach zawartych w katalogu, najłatwiej jest wykorzystać interpreter **bash** i jego wbudowany język skryptowy.
- Pomocne mogą okazać się jeszcze: **vim** lub **nano** (edytory tekstu), **less** (stronicowanie), **column** (formatowanie), **file** (identyfikacja plików), **pr** (przygotowanie plików do druku), **head** (wyświetlenie początkowych linii pliku), **tail** (wyświetlenie końcowych linii pliku), **zip** (pakowanie plików), **unzip** (rozpakowanie plików).
- Proszę o przypomnienie podstawowych poleceń służących do pracy z plikami i katalogami: **ls**, **cp**, **mv**, **rm**, **mkdir**, **rmdir** (będzie lub było na **WSK**).
- Proszę o przypomnienie filozofii pracy ze strumieniami i potokami, operatory: **|**, **<**, **>**, **>>** (będzie lub było na **WSK**).
- Podczas pracy z plikami tekstowymi należy pamiętać o odmiennym znaku końca linii. W przypadku Windows to jest **CR LF** (**\r** i **\n**), a w przypadku GNU/Linux to **LF** (**\n**). W konwersji mogą pomóc: **tr**, **dos2unix**, **unix2dos**.

Zadania

1. Zespoły i narzędzia

- Podział grupy laboratoryjnej na zespoły dwuosobowe.
- Założenie konta w darmowym repozytorium **git**.
- Założenie konta w darmowym systemie edycji **dokumentów on-line**.
- Zapoznanie się z językiem znaczników **Markdown** (np. <https://www.markdownguide.org>).

2. MSYS2

- Zapoznanie się z systemem **MSYS2** (<https://www.msys2.org>).
- Instalacja, konfiguracja MSYS2 (instalacja na dysku lokalnym, prawdopodobnie **d:**).
- Zapoznanie się z rodzajami środowiska MSYS2 (**msys**, **ucrt64**, **clang64**, **mingw64**).
- Zapoznanie się z poleceniem **pacman**, (aktualizacja i zarządzanie pakietami oprogramowania).
- Aktualizacja systemu MSYS2, instalacja dodatkowych pakietów (jeśli nie są już zainstalowane): **vim**, **nano**, **less**, **diffutils**, **zip**, **unzip**, **dos2unix**, **patch**, **imagemagick** (dla środowiska ucrt64 będzie to: **mingw-w64-ucrt-x86_64-imagemagick**).

3. Zagadka

Kolega przysłał plik **zagadka.txt** (**zagadka.zip** na stronie z materiałami) z prośbą: *coś jest nie tak, arkusz pokazuje krzaki, a program w C źle liczy linie*. Bez otwierania pliku w edytorze należy ustalić:

- jakiego typu jest plik i w jakim jest kodowaniu,
- jakie ma znaki końca linii,
- ile ma linii, bajtów i znaków,
- czy zawiera tabulatory, BOM lub znaki spoza ASCII,
- czym kończy się plik.

Dla każdego użytego polecenia i każdej opcji należy w sprawozdaniu podać, **skąd wiadomo**, co robi – fragment strony **man** lub wyniku **--help**. Samo polecenie bez źródła nie jest zaliczane.

Dozwolone polecenia: **file**, **wc**, **head**, **tail**, **od** lub **xxd**, **grep**, **cat**, **man**.

4. Niesforne dane

W pliku **dane.txt** (**dane.zip** na stronie z materiałami) zawarte są dane liczbowe w jednej kolumnie.

```
0
0
0
0,001
0,000999999833333342
0,000159235589171981
...
```

Należy przygotować wersję pliku z danymi w trzech kolumnach, aby można było łatwo importować plik do arkusza kalkulacyjnego i naszkicować wykresy. W pierwszym wierszu należy dodać nazwy kolumn, np.: **x**, **y** i **z**.

```
x      y      z
0      0      0
0,001  0,000999999833333342  0,000159235589171981
...
```

5. Dodawanie poprawek

W trakcie pracy nad programem, w którym wykorzystuje się listę słów zawartą w pliku **lista.txt** (**lista.zip**) powstała nowa, poprawiona wersja listy **lista-pop.txt**. Należy sprawdzić czym się obie listy różnią. Następnie przygotować łąkę pozwalającą na łatwe uaktualnienie listy oryginalnej do listy

aktualnej i krótką informację jak to zrobić. Aby sprawdzić czy poprawki zostały poprawnie dodane, należy obliczyć i porównać sumy kontrolne **MD5** pliku **lista.txt** (po poprawkach) i pliku **lista-pop.txt**.

6. Z CSV do SQL i z powrotem

W pliku **steps-2sql.csv** (**csv.zip**) znajdują się dane w trzech kolumnach z separatorem w postaci średnika wyeksportowane z systemu pomiarowego.

```
time;intensity;steps
1562001120;19;0
1562001180;23;0
...
```

Należy przygotować dane do importu do bazy danych **SQL** w poniższej postaci.

```
INSERT INTO stepsData (time, intensity, steps) VALUES (1562001120, 19, 0);
INSERT INTO stepsData (time, intensity, steps) VALUES (1562001180, 23, 0);
...
```

W drugiej części zadania opracować metodę zamiany skryptu **SQL** (plik **steps-2csv.sql**).

```
INSERT INTO stepsData (dateTime, steps, synced) VALUES (1562004600000, 41, 0);
INSERT INTO stepsData (dateTime, steps, synced) VALUES (1562005200000, 65, 0);
...
```

na plik typu **CSV** w postaci.

```
dateTime;steps;synced
1562004600;41;0
1562005200;65;0
...
```

Należy zwrócić uwagę na to, że zmieniła się dokładność stempla czasu (*Unix/Epoch Timestamp*) z milisekund do sekund co spowodowało konieczność usunięcia trzech ostatnich zer z wartości **dateTime**.

7. Marudny tłumacz

Podczas wdrażania oprogramowania zaszła potrzeba uzupełnienia tłumaczenia oprogramowania z języka angielskiego na język polski. Angielska wersja językowa znajduje się w pliku typu **JSON** o nazwie **en-7.2.json5** (z pliku **tlumacz.zip**):

```
{
  "form.cancel": "Cancel",
  "form.clear": "Clear",
  "form.edit": "Edit",
  ...
}
```

Na prośbę tłumacza należy zdublować wszystkie linie w pliku, żeby ułatwić mu tłumaczenie. Nazwy kluczy muszą być unikatowe, zatem należy dodać komentarz do dodanych linii, nowy plik należy zapisać pod nazwą: **pl-7.2.json5**:

```
{  
  
  // "form.cancel": "Cancel",  
  "form.cancel": "Cancel",  
  
  // "form.clear": "Clear",  
  "form.clear": "Clear",  
  
  // "form.edit": "Edit",  
  "form.edit": "Edit",  
  
  ...  
}
```

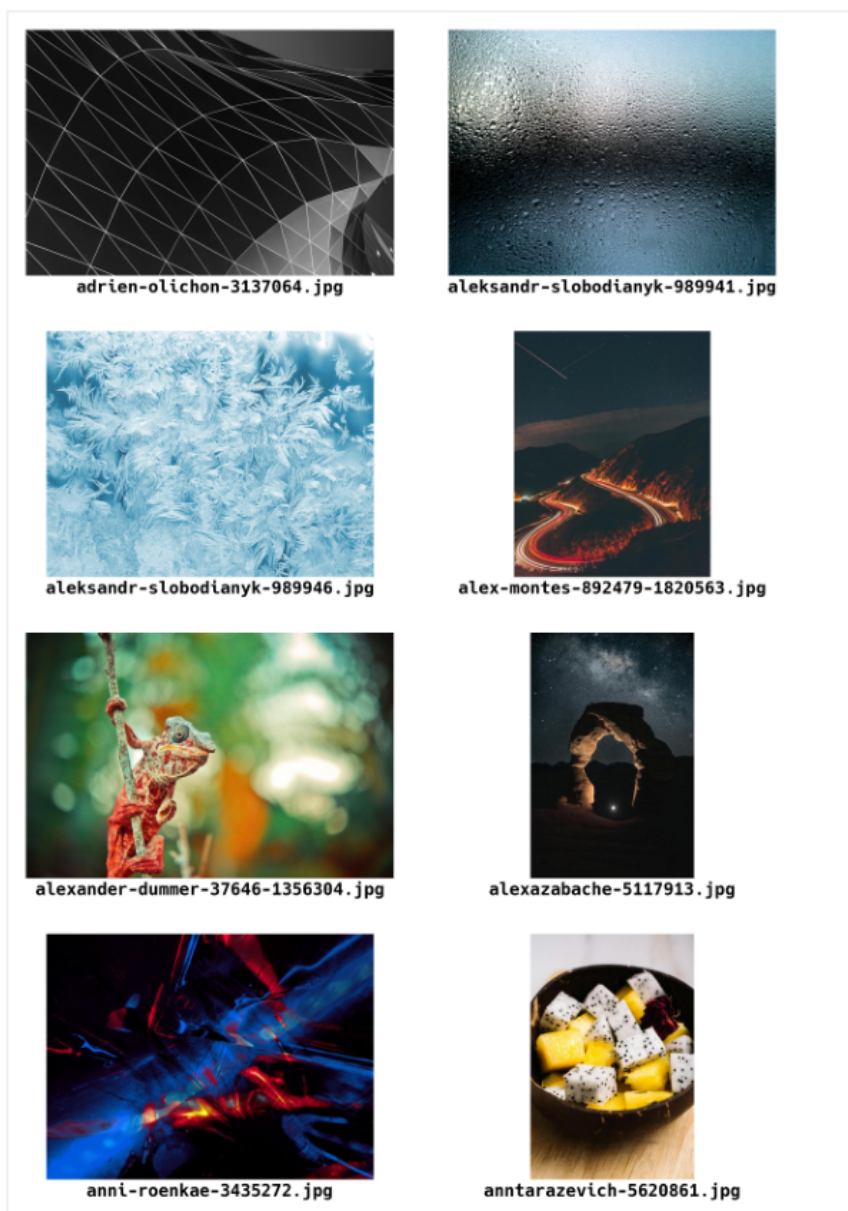
Niestety, w trakcie pracy tłumacza została wydana nowa wersja wdrażanego programu (plik **en-7.4.json5**) i trzeba przygotować dodatkowy plik z frazami do tłumaczenia. Dodatkowy plik **pl-7.4.json5** powinien zawierać tylko nowe napisy nie występujące w poprzedniej wersji.

8. Fotografik gamoń

Kolega fotografik poprosił o pomoc w zmianie formatu i zmniejszeniu rozdzielczości zdjęć przed wysyłką. Z powodu posiadania małego dysku wszystkie jego prace są spakowane w osobnych plikach, całość znajduje się w dwóch plikach (ze względu na ograniczenia serwera): **kopie-1.zip** i **kopie-2.zip**. Pliki należy rozpakować, zmienić format na **JPG** (niektóre), ustawić rozdzielczość **DPI** każdego pliku na **96x96** i wysokość na **720 pikseli**. Całość potem spakować do jednego archiwum **ZIP**.

9. Wszędzie te PDF-y

Okazało się, że wcześniej wspomniany kolega, potrzebuje jeszcze portfolio swoich prac w pliku **PDF**, który będzie można łatwo wydrukować. Z wcześniej sformatowanych plików należy wygenerować dokument **A4**, z ośmioma fotografiami na stronie, po dwie fotografie w czterech wierszach. Dodatkowo trzeba dodać nazwy plików do dokumentu PDF.



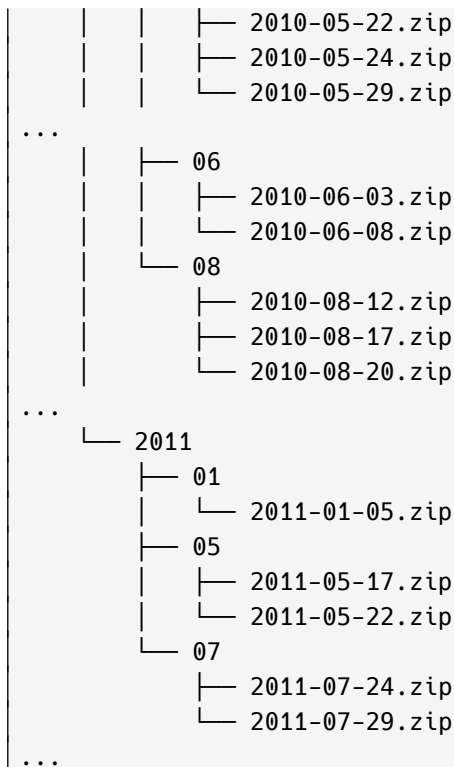
10. Porządki w kopiach zapasowych

Dostaliście Państwo w spadku system plików z kopiami zapasowymi (ze względu na ograniczenia serwera w dwóch plikach: **kopie-1.zip** i **kopie-2.zip**). Wszystkie pliki są w jednym katalogu.

```
.
├─ kopie
│   ├── 2010-05-22.zip
│   ├── 2010-05-24.zip
│   └── 2010-05-29.zip
└─ ...
```

Zadaniem jest uporządkowanie plików kopii w taki sposób, żeby każdy rok miał swój katalog i w każdym roku kopie z jednego miesiąca znajdowały się w osobnych podkatalogach.

```
.
├─ kopie
│   ├── 2010
│   │   └── 05
│   └── ...
```



11. Galeria dla grafika

Kolega grafik poprosił o pomoc w szybkim wykonaniu galerii internetowej z plikami graficznymi. Macie Państwo do dyspozycji szablony galerii ze starego projektu (**galeria.zip**). Należy użyć plików graficznych z zadania nr 7.

W poniższym kodzie **HTML** podkreśleniem zostało zaznaczone miejsce, w którym powinny się znaleźć nazwy plików graficznych. Ta część kodu musi być wielokrotnie wstawiona w szablon galerii.

```
<div class="responsive">
  <div class="gallery">
    <a target="_blank" href="blue.png">
      
    </a>
    <div class="desc">blue.png</div>
  </div>
</div>
```

