Vortaro

Slovník v příkazovém řádku

Program, napsáný v rámci semestrální práce předmětu Programování v jazyku C, je určen k volnému použití pod licencí GPL GNU v.3

Mgr.Jiří Hrbek, 2008

Obsah dokumentace

1 Dokumentace pro uživatele 3
1.1 Důvod vzniku programu 3
1.2 Instalace programu 4
1.2.1 Instalace v prostředí Linux 4
1.2.2 Instalace v prostředí Windows - bez kompilace 6
1.2.3 Instalace v prostředí Windows - včetně kompilace 7
1.2.4 Instalace na ostatních platformách 9
1.3 Použití programu 11
1.3.1 Použití programu v neinteraktivním režimu 11
1.3.2 Použití programu v interaktivním režimu 12
1.3.3 Použití s větším počtem různých datových souborů 12

2 Dokumentace pro programátora 14
2.1 Celkový přehled souborů 14
2.2 Podrobnější popis struktur a funkcí 15
2.2.1 SXLOSILOJ - klíče 15
2.2.2 KONVKODADO - konverze kódování 16
2.2.3 KONVERTO - konverze csv-souboru do dat-souboru 17
2.2.4 LISTO - obousměrný seznam 19
2.2.5 TRADUKADO - překládání 20
2.2.6 MemArangxo - kontrolní systém uvolnění všech alokovaných bloků v haldě 22
2.2.7 Symbolické konstanty a makra 23

3 Rejstřík 25

1 Dokumentace pro uživatele

1.1 Důvod vzniku programu

Od roku 2006 mají esperantisté vládnoucí českým jazykem k volnému použití slovník, rozsahem i důkladností monumentální dílo pana Josefa Hrona[1]. Nejbolestivější slabinou pohodlného používání této práce je jeho forma. Jde totiž o datový soubor tabulkového kalkulátoru, který v době psaní této dokumentace obsahuje úctyhodných 169912 položek. Pouhé načtení takto velkého souboru do tabulkového kalkulátoru trvá tak dlouho, že spolehlivě odradí od občasného použití pro ojedinělý výraz. Vyhledávání jednotlivých výrazů po načtení se na běžně vybaveném počítači pohybuje v řádu několika vteřin, což také nelze označit za vysoký komfort. Pro použití v příkazovém řádku, což je oblíbená forma práce mnoha uživatelů operačních systémů unixového typu, také není forma datového souboru pro tabulkový kalkulátor nejvhodnější.

Z těchto důvodů jsem se rozhodl napsat program, který výše zmíněné nevýhody odstraní. Svůj program jsem nazval **vortaro**[2] a při jeho tvorbě jsem preferoval následující požadavky:

- **přenositelnost** - program je vytvořen v ANSI C, takže ho lze přeložit pro prakticky libovolnou platformu. Pro nejrozšířenější desktopové platformy (Windows/Linux) je kompilace, instalace i odinstalace usnadněna jednoduchými skripty.
- **rychlost** - program si na základě datového souboru pro tabulkový kalkulátor vytvoří vlastní oindexovaný soubor, pomocí kterého bude vyhledávací služby poskytovat v maximální rychlosti.
- **integrace v prostředí příkazového řádku** - program lze pouštět jednorázově, kdy vstup vložíme pomocí parametrů a výstup (tedy překlad výrazu) můžeme nasměrovat na obrazovku, do souboru, nebo do dalšího programu - jak je běžné u jiných příkazů konsole. Možná je také práce v interaktivním režimu, kdy můžeme listovat nejbližším okolím vyhledaného hesla.

[1] k dispozici **http://esperanto.wz.cz/**

[2] v esperantu výraz pro 'slovník'

1.2 Instalace programu

1.2.1 Instalace v prostředí Linux

V této části předpokládám, že uživatel disponuje některou z běžných distribucí operačního systému Linux, s nainstalovaným kompilátorem **gcc**, programem **make**, **unzip** a příkazovým interpretem typu **BASH**. Při instalaci prosím sledujte následující postup:

- **Dekomprimace** Rozbalte stažený soubor do libovolné složky, ve které máte právo čtení i zápisu:

  ```
  unzip vortaro.zip
  ```

  a vstupte do rozbalené složky Vortaro.

  ```
  cd Vortaro
  ```

- **Kompilace** Program si zkompilujte pomocí příkazu:

  ```
  make -f alLin kompili
  ```

  Pokud vše proběhne v pořádku, měla by být složka doplněna několika soubory s koncovkou '.o' a hlavně souborem _vortaro. Tento spustitelný soubor poskytuje veškerou funkčnost programu. Nezvyklé je na něm pouze to, že je třeba při spouštění uvádět celou absolutní cestu.[3] Příklad:

  ```
  \home\uzivatel\bin\_vortaro -h
  ```

  Vyhnul jsem se tak nutnosti používat globální proměnné systému. Použití programu to nijak nekomplikuje, pouze se místo přímého spuštění programu bude spouštět krátký skript, který tento požadavek zajistí.

- **Instalace** Nyní program nainstalujeme. Před tímto krokem je třeba zvážit, kam program umístíte. Je totiž nutné mít na dané místo právo zápisu a také by mělo být umístěno v proměnné `PATH`. Doporučuji se tedy před tímto krokem přihlásit jako `root`. Instalaci provedeme buď opět přes příkaz `'make'`:

  ```
  make -f alLin instali
  ```

  Nebo přímo spuštěním skriptu

  ```
  ./instvort.lin
  ```

[3] zjišťuje si tak totiž umístění svého defaultního datového souboru

Budete dotázáni, kde chcete mít program s daty umístěn:

```
Kie vi volas vortaro havi?
Situo devas en parametro PATH esti. [/usr/bin]
```

Pokud jste s umístěním `/usr/bin` spokojeni, stačí stisknout `Enter`, jinak zadejte obdobnou cestu. Program by Vás měl bez chybových hlášení poinformovat, že je nainstalován a že je možné přistoupit ke konverzi hlavního datového souboru:

```
Aplikajxo 'vortaro' estas instalita.
Nun vi povas konvertu csv-dosiero per elekto '-k'... - legu helpon:
('vortaro -h').
```

Na Vámi zvoleném umístění se nyní nachází spustitelný skript `vortaro` a složka `vortdosiero`, do které byl zkopírován soubor _vortaro

- **Konverze datového souboru** Ve složce `Vortaro` se nachází soubor `EspSlovnik.csv`, který je aktuální v době psaní této dokumentace. Stáhl jsem ho ze stránky **http://esperanto.wz.cz/** a pomocí OpenOffice 2.0 uložil ve formátu CSV. V případě, že budete vytvářet svůj vlastní CSV soubor, dejte pozor, aby byl uložen ve formátu UTF-8. Pokud jste stále ve složce `Vortaro`, můžete konverzi datového souboru provést jednoduše jedním z příkazů:

  ```
  vortaro -ku EspSlovnik.csv
  vortaro -ki EspSlovnik.csv
  ```

  Kterou z voleb použít záleží na češtině, kterou používá Vaše distribuce. První volba předpokládá UTF-8, druhá ISO-8859-2[4]. Pokud si nejste jisti, zvolte první možnost a v případě problémů se zobrazením interpunkce, zkuste později druhou. Ve druhém případě bude esperantská diakritika nahrazena spřežkami (ĉ,ĥ,ĝ, ŭ... -> cx, hx, gx, ux...).

  Konverze nějakou dobu trvá, pokud proběhne úspěšně, oznámí Vám to nápis:

  ```
  Konverto finigxis...
  ```

  Ve složce `vortdosiero` by se nyní měl nacházet datový soubor s názvem `espvortaro.dat`.

- **Případná odinstalace** Tím je program `vortaro` připraven k používání. Ve složce `Vortaro` kromě této dokumentace nyní přítomen také skript `malinstvort.lin`, jehož spuštěním můžete `vortaro` ze svého systému odinstalovat. Ostatní již s velkou pravděpodobností potřebovat nebudete.

[4] známo také jako ISO Latin 2

1.2.2 Instalace v prostředí Windows - bez kompilace

V této části předpokládám, že uživatel disponuje některou z 32-bitovou verzí operačního systému Windows.

- **Dekomprimace** Rozbalte stažený soubor a výslednou složku `Vortaro` přesuňte například do kořenového adresáře složky C:\. Otevřete `Příkazový řádek` a vstupte do rozbalené složky Vortaro:

  ```
  c:
  cd \Vortaro
  ```

- **Instalace** Program je třeba instalovat na takové místo v adresářové struktuře, které je uvedeno v proměnné %PATH%. Které to jsou můžete zjistit příkazem:

  ```
  echo %PATH%
  ```

  Dále musíte mít ve vybraném umístění právo zápisu. Upravování proměnné %PATH% a nastavování práv uživatelů překračuje cíl této dokumentace. Pokud máte vybráno, spusťte dávkový soubor:

  ```
  ./instrapide.win.bat
  ```

  Budete dotázáni, kde chcete mít program s daty umístěn:

  ```
  Kie vi volas vortaro havi?
  (Situo devas en parametro PATH esti):
  ```

  Předpokládám, že odpovíte například:

  ```
  c:\windows
  ```

  Program by Vás měl bez chybových hlášení poinformovat, že je nainstalován a že je možné přistoupit ke konverzi hlavního datového souboru:

  ```
  Aplikajxo 'vortaro' estas instalita.
  Nun vi povas konvertu csv-dosiero per elekto '-k'... - legu helpon:
  ('vortaro -h').
  ```

  Na Vámi zvoleném umístění se nyní nachází spustitelný skript

  ```
  c:\windows\vortaro.bat
  ```

  a složka

  ```
  c:\windows\vortdosiero
  ```

do které byl zkopírován soubor

```
c:\windows\vortdosiero\_vortaro.exe
```

- **Konverze datového souboru** Ve složce `Vortaro` se nachází soubor `EspSlovnik.csv`, který je aktuální v době psaní této dokumentace. Stáhl jsem ho ze stránky **http://esperanto.wz.cz/** a pomocí OpenOffice 2.0 uložil ve formátu CSV. V případě, že budete vytvářet svůj vlastní CSV soubor, dejte pozor, aby byl uložen ve formátu UTF-8. Pokud jste stále ve složce `Vortaro`, můžete konverzi datového souboru provést jednoduše příkazem:

  ```
  vortaro -kc EspSlovnik.csv
  ```

  V příkazovém řádku se bohužel používá 'čeština' známá jako CP852[5], kde na rozdíl od UTF-8 nelze zobrazovat českou a esperantskou diakritiku zároveň. Během konverze a indexování CSV souboru bude tedy esperantská diakritika nahrazena spřežkami (ĉ,ĥ,ĝ, ŭ... -> cx, hx, gx, ux...).

  Konverze nějakou dobu trvá, pokud proběhne úspěšně, oznámí Vám to nápis:

  ```
  Konverto finigxis...
  ```

  Ve složce `c:\windows\vortdosiero` by se nyní měl nacházet datový soubor s názvem `espvortaro.dat`.
- **Případná odinstalace** Tím je program `vortaro` připraven k používání. Ve složce `Vortaro` kromě této dokumentace nyní přítomen také skript `malinstvort.win.bat`, jehož spuštěním můžete `vortaro` ze svého systému odinstalovat. Ostatní již s velkou pravděpodobností potřebovat nebudete.

1.2.3 Instalace v prostředí Windows - včetně kompilace

V této části předpokládám, že uživatel disponuje některou aktuální verzí Windows a má nainstalované vývojové prostředí Microsoft Visual C++ 2008 Express Edition

- **Dekomprimace** Rozbalte stažený soubor a výslednou složku `Vortaro` přesuňte například do kořenového adresáře složky C:\. Otevřete `Příkazový řádek` a vstupte do rozbalené složky Vortaro:

  ```
  c:
  cd \Vortaro
  ```

[5] nebo také PC Latin 2

- **Kompilace** Program si zkompilujte pomocí příkazu:

```
nmake -f alWin kompili
```

Pokud vše proběhne v pořádku, měla by být složka doplněna několika soubory s koncovkou '.obj' a hlavně souborem _vortaro.exe. Tento spustitelný soubor poskytuje veškerou funkčnost programu. Nezvyklé je na něm pouze to, že je třeba při spouštění uvádět celou absolutní cestu.[6] Příklad:

```
C:\Vortaro\_vortaro.exe -h
```

Vyhnul jsem se tak nutnosti používat globální proměnné systému. Použití programu to nijak nekomplikuje, pouze se místo přímého spuštění programu bude spouštět dávkový soubor, který tento požadavek zajistí.

- **Instalace** Program je třeba instalovat na takové místo v adresářové struktuře, které je uvedeno v proměnné %PATH%. Které to jsou můžete zjistit příkazem:

```
echo %PATH%
```

Dále musíte mít ve vybraném umístění právo zápisu. Upravování proměnné %PATH% a nastavování práv uživatelů překračuje cíl této dokumentace. Pokud máte vybráno, použijte příkaz:

```
nmake -f alWin instali
```

Budete dotázáni, kde chcete mít program s daty umístěn:

```
Kie vi volas vortaro havi?
(Situo devas en parametro PATH esti):
```

Předpokládám, že odpovíte například:

```
c:\windows
```

Program by Vás měl bez chybových hlášení poinformovat, že je nainstalován a že je možné přistoupit ke konverzi hlavního datového souboru:

```
Aplikajxo 'vortaro' estas instalita.
Nun vi povas konvertu csv-dosiero per elekto '-k'... - legu helpon:
('vortaro -h').
```

Na Vámi zvoleném umístění se nyní nachází spustitelný skript

[6] zjišťuje si tak totiž umístění svého defaultního datového souboru

`c:\windows\vortaro.bat`

a složka

`c:\windows\vortdosiero`

do které byl zkopírován soubor

`c:\windows\vortdosiero\_vortaro.exe`

- **Konverze datového souboru** Ve složce `Vortaro` se nachází soubor `EspSlovnik.csv`, který je aktuální v době psaní této dokumentace. Stáhl jsem ho ze stránky **http://esperanto.wz.cz/** a pomocí OpenOffice 2.0 uložil ve formátu CSV. V případě, že budete vytvářet svůj vlastní CSV soubor, dejte pozor, aby byl uložen ve formátu UTF-8. Pokud jste stále ve složce `Vortaro`, můžete konverzi datového souboru provést jednoduše příkazem:

  `vortaro -kc EspSlovnik.csv`

  V příkazovém řádku se bohužel používá 'čeština' známá jako CP852[7], kde na rozdíl od UTF-8 nelze zobrazovat českou a esperantskou diakritiku zároveň. Během konverze a indexování CSV souboru bude tedy esperantská diakritika nahrazena spřežkami (ĉ,ĥ,ĝ, ŭ... -> cx, hx, gx, ux...).

  Konverze nějakou dobu trvá, pokud proběhne úspěšně, oznámí Vám to nápis:

  `Konverto finigxis...`

  Ve složce `c:\windows\vortdosiero` by se nyní měl nacházet datový soubor s názvem `espvortaro.dat`.
- **Případná odinstalace** Tím je program `vortaro` připraven k používání. Ve složce `Vortaro` kromě této dokumentace nyní přítomen také skript `malinstvort.win.bat`, jehož spuštěním můžete `vortaro` ze svého systému odinstalovat. Ostatní již s velkou pravděpodobností potřebovat nebudete.

1.2.4 Instalace na ostatních platformách

V této části předpokládám, že uživatel nemůže využít předchozích popisů a chce program instalovat na zcela odlišnou platformu.

- **dekomprimace** - rozbalte stažený soubor a přesuňte se do výsledné složky `Vortaro`. Zde kromě jiného najdete zdrojový kod programu (kodado.c, kodado.h, listo.c, listo.h, malloko.c, malloko.h, sxlosiloj.c, sxlosiloj.h, vortaro.c, vortaro.h). Je

[7] nebo také PC Latin 2

potřeba ho zkompilovat libovolným kompilátorem akceptujícím ANSI C. Pokud je na vaší platformě dostupný nástroj usnadňující kompilaci - `make/nmake`, můžete využít souborů makefile, které najdete ve složce pod názvem `alLin/alWin`.

- **instalace** - výsledný spustitelný soubor je třeba spouštět včetně kompletní cesty, kde je umístěn a to i v případě, že se zrovna ve stejné složce nacházíte. Je proto vhodné vytvořit na Vaší platformě spouštěcí skript, s podobnou funkcí, jako má onen na platformě Lin:

```
#! /bin/bash
# komentar: program se nachazi v /usr/bin/
# komentar: znaky $* jsou parametry predane skriptu

/usr/bin/_vortaro $*
```

  a ten umístit na místo, ve kterém systém spouštěcí skripty očekává. Jako kontrolu, že je Váš program spustitelný, ho zkuste spustit s parametrem '-h'. Pokud se Vám na konsoli objeví stručný popis použití v esperantu, je vše v pořádku.

- **Konverze datového souboru** V rozbalené složce `Vortaro` se nachází soubor `EspSlovnik.csv`, který je aktuální v době psaní této dokumentace. Stáhl jsem ho ze stránky **http://esperanto.wz.cz/** a pomocí OpenOffice 2.0 uložil ve formátu CSV. V případě, že budete vytvářet svůj vlastní CSV soubor, dejte pozor, aby byl uložen ve formátu UTF-8. Pokud jste stále ve složce `Vortaro`, program je již spustitelný např. příkazem `vortaro`, můžete konverzi datového souboru provést jednoduše jedním z příkazů:

```
vortaro -k EspSlovnik.csv
vortaro -kc EspSlovnik.csv
vortaro -ki EspSlovnik.csv
vortaro -kw EspSlovnik.csv
```

  Který použít, záleží na používaném kódování 'češtiny' ve Vašem OS. Předchozí příkazy přizpůsobí datový soubor pořadě těmto kodováním: UTF-8, ISO-8859-2[8], CP852[9], Windows-1250.

  Pokud se Vám konverze povede, měl by se v místě, kde se nachází Vámi zkompilovaný program, objevit soubor `espvortaro.dat`. V takovém případě jste instalaci zřejmě zvládli a můžete pokračovat v čtení o použití programu...

[8] ISO Latin 2
[9] PC Latin 2

1.3 Použití programu

Vyhledávání hesel je podřízeno struktuře slovníku, který obsahuje u každého hesla 9 položek:

- ŝlosilo1[10] - obsahuje seznam klíčů v esperantu, které mají k danému heslu nějaký vztah. Každý klíč začíná a končí tečkou, takže tato položka může obsahovat například:

  `.paciga jugxisto. .pacojugxisto.`

- ŝlosilo2[10] - obsahuje seznam klíčů v češtině, které mají k danému heslu nějaký vztah. každý klíč začíná a končí čárkou, takže tato položka může obsahovat například:

  `,smířit se, ,usmířit se,`

- vortradiko[11] - například `pac` pro výrazy `pacema, paciga, pacigi...`
- esperante (výraz v esperantu)
- fonto[12] - například `hovor, hud, křest ...`
- ĉeĥe (výraz v češtině)
- noto[13] - obvykle upřesnění použití
- komparu[14] - odkaz na jiné výrazy, které mají k současnému vztah

Program `vortaro` prohledává pouze v prvních dvou položkách[15] a jeho výstupem je zbývajících 7. Stává se, že stejný klíč je uveden u více hesel, takže výstupem programu je často více takových sedmic. Příklady možných vyhledávaných výrazů:

1.3.1 Použití programu v neinteraktivním režimu

Pro tento režim voláme program s přepínačem '-t':

```
vortaro -t ,ahoj, ijo.
```

[10] ŝlosilo=klíč
[11] vortradiko = kořen slova
[12] fonto = zdroj
[13] noto = poznámka
[14] komparu = porovnej
[15] tzn. `ŝlosilo1, ŝlosilo2`

výraz	**vyhledá všechny položky,**
.lingvo.	u kterých je esperantský klíč `lingvo`
,jazyk,	u kterých je český klíč `jazyk`
ejo.	jejichž esperantský klíč končí na `-ejo`
,vy	jejichž český klíč začíná předponou `vy-`
elo	jejichž klíč (český nebo esperantský) obsahuje `elo`

Tabulka 1.1 příklady vyhledávání

Program si načte z datového souboru `vortdosiero/espvortaro.dat` oindexované klíče, vyhledá mezi nimi všechny české, které obsahují 'ahoj' a všechny esperantské, které končí na '-ijo'. Nalezené vytiskne na standartní výstup, uvolní oindexované klíče a skončí. Pro soustavnější vyhledávání je výhodnější interaktivní režim...

1.3.2 Použití programu v interaktivním režimu

Pro tento režim voláme program s jediným přepínačem '-ti':

```
vortaro -ti
```

Program se vás zeptá na slovo pro přeložení (`vorto por traduki`). Místo slova můžete stisknout ctrl+f[16], nebo ctrl+q a po následném `enter` se program ukončí. Jinak vyhledá položku s nejbližším klíčem, pomocí kláves 'h'/'l' listovat nejbližším okolím, přes volbu 'n' vyhledávat nový výraz, nebo opět ukončit program přes ctrl+f/q.

1.3.3 Použití s větším počtem různých datových souborů

Je možné si překonvertovat více různých CSV souborů a používat libovolný z nich v ne/interaktivním režimu. V takovém případě použijte pro konverzi následující zápis:

```
vortaro -k soubor.csv novysoubor.dat
```

Součástí názvů souborů může být samozřejmě kompletní cesta v adresářové struktuře. V uvedeném příkladě se soubor `'novysoubor.dat'` vytvoří v aktuální složce. Přinutit program `vortaro`, aby použil místo standartního `'espvortaro.dat'` ve složce `vortdosiero` - soubor `novysoubor.dat'`, lze pomocí přepínače '-td'/'-tid'[17]:

[16] od slova fino = konec
[17] pro interaktivní/neinteraktivní režim

```
vortaro -td novysoubor.dat ,ahoj, ijo.
vortaro -tid novysoubor.dat
```

Za parametrem '-td'/'-tid' lze samozřejmě opět jako jméno souboru uvést celou cestu. Pro pohodlné používání je vhodné si pro časté používání různých datových souborů připravit různě pojmenované spouštěcí skripty, podle šablony již existujícího `vortaro`.

2 Dokumentace pro programátora

2.1 Celkový přehled souborů

Celý program je rozdělen do pěti souborů:

- **vortaro** - v souboru `vortaro.c` se nachází počáteční funkce `main`, která dle parametrů spouští hlavní funkce programu. Dále obsahuje základní struktury a funkce, které řídí dvě základní funkce programu - konverze a překlad.
- **sxlosiloj** - obsahuje objekt[18], jehož funkcí je udržovat v paměti RAM klíče - tedy vyhledávané položky. U každé z nich je i odkaz na ostatní data, která je po vyhledání nutné zobrazit, ale pro svou obsáhlost zůstávají po celý běh v datovém souboru a načítány jsou jen jednotlivě v případě úspěšného vyhledání. Základní funkce objektu `SXLOSILOJ` jsou:
  - ⋆ **aldonuSxlosilo** - uložení konkrétního klíče
  - ⋆ **rangxuSxlosiloj** - seřazení klíčů pro budoucí snadnější vyhledávání
  - ⋆ **konservuSxlosiloj** - uložení seřazených klíčů do datového souboru
  - ⋆ **surekranigiSxlosiloj** - načtení seřazených klíčů z datového souboru
- **kodado** - obsahuje objekt, jehož funkcí je překládat kódování UTF-8 do jedné z 'čeština':
  - ⋆ Windows-1250
  - ⋆ ISO-8859-2[19]
  - ⋆ CP852[20]
- **listo** - obsahuje objekt, který poskytuje základní funkce běžného obousměrného seznamu. Tento objekt neměl být původně součástí tohoto projektu, proto disponuje mnoha funkcemi, které přímo s programem vortaro nesouvisí. Proměnil jsem je v komentáře, pro případ, že by se později hodily.
- **malloko** - malloko znamená opak 'lokálního' - tedy globální. Tento soubor obsahuje vše, co je potřeba mít dostupné ve všech ostatních. Především se jedná o vlastní systém kontroly alokované a uvolněné paměti, který se zapíná, když je definována symbolická konstanta DEBUG. Bohužel nelze funkci `mallinfo()` využívat v prostředí Win, takže pro správnou činnost v DEBUG modu by bylo

[18] objekt - tedy strukturu + metody
[19] známo také jako ISO Latin 2
[20] známo tako jako PC Latin 2

nutné vytvořit paraelní systém kontroly alokace paměti i pro tuto platformu. To jsem ale nepovažoval za nutné, protože kladný výsledek kontroly na platformě Linux je platný i pro všechny ostatní platformy, na kterých program poběží.

Kromě těchto pěti základních souborů je součástí projektu také několik skriptů, usnadňujících kompilaci, instalaci a funkčnost na dvou hlavních platformách:

- **alLin, alWin** - Makefile soubory pro kompilaci a instalaci programu
- **instvort.lin, instvort.win.bat** - skript, který se zeptá na místo v adresářové struktuře, na kterém potom vytvoří složku vortdosiero, do ní zkopíruje zkompilovaný program a vytvoří spouštěcí skript.
- **instalurapide.win.bat** - pro snadnější instalaci uživatelům OS Win, je součástí již zkompilovaný binární soubor pro tuto platformu. Při instalaci bez kompilace tento dávkový soubor zkopíruje zmíněný binární soubor na vybrané umístění a vytvoří spouštěcí skript.
- **purigu.lin, purigu.win.bat** - odstraní všechny soubory po předchozí kompilaci, je tedy možné pokusit se o novou.
- **EspSlovnik.csv** - soubor dat pro naplnění slovníku po instalaci programu.

2.2 Podrobnější popis struktur a funkcí

2.2.1 SXLOSILOJ - klíče

Hlavní funkcí tohoto objektu (struktury `SXLOSILOJ` a s ní spolupracující funkce) je spravovat krátké textové řetězce[21], tzn. udržovat je v paměti, řadit je, ukládat do souboru a opět je ze souboru načíst. Každý klíč uložený do této struktury souvisí s určitými externími daty na disku, takže za každým klíčem je uloženo také číslo `long int`, které je offsetem souvisejících externích dat v datovém souboru.

Jedná se v podstatě o dvě dynamická pole - jedno na text, druhé na čísla `long int`:

- `char *sxlosteksto`
- `long int *sxlosnombro`

V případě požadavku na uložení dalšího klíče (`aldonuSxlosilo`), je do `*sxlosteksto` uložen klíč spolu s offsetem (`long int`) externích dat v datovém souboru

[21] v našem případě klíče

a do `*sxlosnombro` je uložen offset daného klíče od adresy `sxlosteksto` - tedy od počátku textové oblasti. V případě, že bude jedna nebo druhá oblast zaplněna, je alokována oblast větší o symbolickou konstantu `tdimensio2/ ndimensio2` a pomocí standartní funkce `memcpy` dosavadní data překopírována. Vedle popsané funkce `aldonuSxlosilo` je funkce dalších souvisejících funkcí následující:

- **SXLOSILOJ *inicSxlosiloj(void)** - vytvoří prázdnou strukturu `SXLOSILOJ`, alokuje počáteční místo pro obě dynamická pole o velikosti (`tdimensio1/ndimensio1`). Dále jsou inicializována ukazovátka nejbližší volné oblasti v obou polích (`tekstmntr/nombrmntr`) a proměnné udržující aktuální velikost obou polí (`tekstdimensio/nombrdimensio`).

- **rangxiordigi,rangxu,rangxuSxlosiloj** - tyto tři funkce zajišťují seřazení klíčů pomocí metody merge-sort. Řadí se vlastně pouze čísla v poli `*sxlosnombro`. Před počátkem řazení je počátek textové oblasti `sxlosteksto` uložen do globální proměnné `rangxitaZono`, protože podle textových řetězců v této oblasti je zjišťováno správné pořadí čísel v oblasti `*sxlosnombro` - viz.použití funkce `strcmp` v `rangxiordigi`. Vyhýbám se tak zbytečnému předávání dalšího parametru.

- **konservuSxlosiloj(FILE *f)** - tato funkce slouží k uložení obou dynamických polí do souboru `f`. Nejprve se uloží velikost pole `*sxlosnombro` a pak samotné pole `*sxlosnombro`. Následuje to samé s polem `*sxlosteksto`.

- **surekranigiSxlosiloj(FILE *f)** - načte ze souboru obě dynamická pole. Tentokrát je již alokována paměť přesně na jejich velikost a přidávání dalších klíčů se již nepředpokládá, takže jsou ukazovátka `-mntr` již zbytečná.

- **fermuSxlosiloj(SXLOSILOJ **sxl)** - uvolní veškerou paměť struktury `**sxl` a nastaví ji na `NULL`

2.2.2 KONVKODADO - konverze kódování

Slouží k nahrazení českých a esperantských znaků s diakritikou v kódování UTF-8, jejich náhradama v kódování Windows-1250, ISO-8859-2, nebo CP852. Tento objekt je využíván hlavně z funkce `ellaboruCSV` v souboru `vortaro.c`.

Struktura `KONVKODADO` obsahuje dva ukazatele, každý na oblast 83 znaků `char -> char *de; char *en`. V oblasti `*de` jsou poskládány dvojice znaků, které mají být v kódování UTF-8 nahrazovány, v oblasti `*en` jsou na odpovídajících pozicích dvojice, které je nahradí. Obě oblasti jsou ukončeny `'\0'`, takže na ně lze aplikovat standartní funkce na zpracování řetězců. V případě, že se dva znaky nahrazují za jeden, musí být druhým znakem v oblasti `*en` - znak `'\0'`. Pokud je při nahrazování druhý znak `'\0'`, funkce `konvertuKodado` jej přeskočí.

- **KONVKODADO* inicKodado(char konverto)** - tato funkce vytvoří strukturu `KONVKODADO` a podle parametru `konverto` ji připraví na konverzi do příslušného kódování.

- **char* konvertuKodado(KONVKODADO *kiel, char *kio)** - podle nastavené konverze `*kiel` vytvoří nový řetězec, který je výstupem této funkce. Paměť alokována pro ukazatel `*kio` bude uvolněna.

- **void neniiguKodado (KONVKODADO **kiu)** - odstraní objekt `**kiu` z paměti a do ukazatele `*kiu` přiřadí `NULL`.

2.2.3 KONVERTO - konverze csv-souboru do dat-souboru

Tento objekt lze nalézt v souborech `vortaro`. Struktura `KONVERTO` obsahuje odkaz na vstupní csv-soubor, na výstupní dat-soubor, znak `kodado` označující, jaká konverze kódování se bude používat (parametr pro výše zmíněnou funkci `inicKodado`) a konečně odkaz na dvě struktury `SXLOSILOJ`. Do jedné z nich budou klíče vkládány obráceně[22]. Důvodem je požadavek na rychlou reakci programu při vyhledávání slov podle koncovky[23]. Související funkce k této struktuře:

- **KONVERTO* inicKonverto (char *csvNomo, char *datNomo, char kodado)** - vytvoří novou strukturu `KONVERTO`, u které zinicializuje oba ukazatele na `SXLOSILOJ`, soubory připraví na čtení/zápis a do DAT-souboru zapíše jedno číslo typu `long int`. Nyní je ještě jedno jaké, později bude přepsáno offsetem, o který bude třeba se posunout, abychom přeskočili externí data a dostali se k uloženým strukturám `SXLOSILOJ`.

- **void ellaboruCSV(KONVERTO *konverto)** - funkce určená pro zpracování csv-souboru. Vstupem je ukazatel na zinicializovanou strukturu `KONVERTO`, to znamená, že soubory jsou již otevřeny a připraveny pro čtení/zápis. Funkce čte soubor-csv po řádcích, každý řádek se může skládat i z několika záznamů[24] a každý z faktů zase z devíti (`NPARTF`) textových řetězců. Ve funkci se každý řádek do těchto řetězců rozděluje, první dva (podle kterých se bude vyhledávat) jsou vždy uloženy do struktur `konverto->sercxsxlosiloj, konverto->reasercxsxlosiloj`, dalších sedm je uloženo do dat-souboru (viz. část funkce s příkazem `switch`). Funkce ellaboruCSV používá několik pomocných funkcí:

  - **char* reiguVorton(const char* vorto)** - funkce podle textového řetězce vytvoří další, s obráceným pořádkem písmen. Takto obrácené řetězce jsou

[22] tzn. každý od posledního písmene po první, např. ahoj -> joha
[23] například 'traduko -t eco.'
[24] rozuměj faktů - `FAKTO`

ukládány do struktury `konverto->reasercxsxlosiloj`, pomocí merge-sort seřazeny a slouží k rychlému vyhledání hesel podle koncovky.

- ⋆ **char* partigiSxlosiloj(char **teksto)** - je volána z funkce ellaboruCSV pro první dva řetězce struktury **FAKTO**: (**sxlosilo1, sxlosilo2**). Každá z nich totiž může být složena z několika klíčů, kde každý z nich je ohraničen tečkami (pro esperantské), či čárkami (pro české). Tato funkce za první z nich vloží ukončovací `'\0'`, ukazatel `*teksto` posune za tuto `'\0'` a vrátí ukazatel na jeho první znak. Když už žádný další klíč v řetězci není, vrátí NULL. Je tedy možné ji volat opakovaně a jednotlivé klíče ukládat do struktur `SXLOSILOJ`.
- ⋆ **char* ceteraLinio(FILE *f, char **linio, char **s, char *cet, KONVKODADO *kod)** - s touto funkcí jsem při prvotním návrhu vůbec nepočítal. Její potřeba vznikla v okamžiku, kdy jsem zjistil, že v csv-souboru může být jeden záznam (`FAKTO`) rozložen do několika řádků (`linio` ve funkci `ellaboruCSV`). Voláním této funkce je zvětšen buffer `linio` pro řádku a načtena k němu řádka následující. Po překódování do správné 'češtiny' jsou obě řádky spojeny, jako by tvořily řádku jedinou.

Přiznám se, že na funkci ellaboruCSV nejsem ani trochu pyšný. Její struktura byla tak dlouho doplňována a opravována, až z ní vznikl dlouhý nepřehledný celek. Zřejmě jde o první místo, které by si zasloužilo přepsat, nicméně funguje spolehlivě a rychle...

- **void donuKonverto(KONVERTO *k)** - tato funkce kompletuje všechny nezbytné akce pro konverzi csv->dat do jednoho celku:
  - ⋆ **ellaboruCSV** - zpracování csv-souboru, naplnění obou struktur `SXLOSILOJ`, uložení všech externích struktur do dat-souboru.
  - ⋆ **rangxuSxlosiloj** - seřazení obou struktur `SXLOSILOJ`.
  - ⋆ **konservuSxlosiloj** - uložení obou struktur `SXLOSILOJ` do dat-souboru, tedy hned za externí data uložená funkcí `ellaboruCSV`. Pozice jejich počátku uložena do `montrilo`.
  - ⋆ **uložení pozice** - na začátku dat-souboru se uloží pozice, kde lze nalézt uložené struktury `SXLOSILOJ`.

- **void konvertu(char *tCSVnomo, char *tDATnomo, char kodado)** - nejvyšší funkce v procesu konverze csv->dat. Pokud je `tDATnomo` nastaveno na NULL, nastaví se do něj `defaultDATsituo`, tedy název souboru `defaultDATnomo`, umístěný ve stejné složce jako náš spuštěný program. Na tento soubor se bude program obracet, pokud nebude výslovně uvedeno jinak[25]. Pak proběhnou tři základní akce procesu konverze - `inicKonverto` (inicializace struktury

[25] přepínačem 'd' lze program nasměrovat na jiný než defaultní soubor

KONVERTO), donuKonverto a fermuKonverto, která uvolní strukturu KONVERTO z paměti.

2.2.4 LISTO - obousměrný seznam

Je objekt, skládající se ze dvou struktur a několika funkcí. První struktura ELEMENT_LISTO představuje prvek v seznamu. Obsahuje ukazatel na následující (sekva), předcházející (antauxa) a ukazatel na data (void *data). Druhá struktura představuje vlastní seznam, obsahuje ukazatel na první prvek v seznamu (unua), na poslední (lasta) a ukazatel na aktuální prvek (montrilo).

- **LISTO* inicListo (void)** - vytvoří a zinicializuje novou strukturu LISTO.
- **int sekvaListo(LISTO *l) / auntauxaListo** - posun vnitřního ukazatele montrilo na další/předchozí prvek. Vrací 0, pokud je parametr l nastaven na NULL
- **void sekvaRingoListo(LISTO *l) / antauxaRingoListo** - jako sekvaListo/antauxaListo, jen pokud dojde na konec/začátek seznamu, přejde opět na začátek/konec.
- **int setListo(LISTO *l,long int wh)** - nastaví absolutní pozici vnitřního ukazatele montrilo. Například setListo(l,1) - nastaví ukazatel na první prvek v seznamu.
- **void* nulListo(LISTO *l, ELEMENT_LISTO *kiu)** - vyřadí ze seznamu prvek, na který ukazuje parametr kiu a ukazatel na jeho data se stane návratovou hodnotou funkce.
- **void* nulTutaListo(LISTO *l)** - volá opakovaně nulListo pro první prvek v seznamu, dokud není prázdný. Na vrácená data je volána funkce free, takže by data neměla obsahovat další ukazatele. V našem programu je LISTO používán ukládání čísel long int (viz.funkce lantaTraduko v souboru vortaro.c), takže tuto funkci lze použít.
- **void* leguMontriloListo(LISTO *l)** - vrátí data[26], která jsou přístupná přes ukazatel montrilo. ELEMENT_LISTO spolu s daty ponechá v seznamu.
- **void neniiguListo(LISTO **l)** - odstraní celý seznam z paměti, do ukazatele *l je přiřazen NULL.
- **cxuMalplenaListo(l)** - makro, které jak již název napovídá, vrací nenulovou hodnotu (pravdu) pokud není LIST *l prázdný.

[26] v našem programu jde o offset, číslo typu long int

2.2.5 TRADUKADO - překládání

Úkolem objektu[27] je vlastní proces vyhledávání požadovaných hesel a oznamování výsledku uživateli.

Struktura `TRADUKADO` obsahuje především ukazatel na datový soubor, ve kterém bude hledat (`DATdosiero`), dva ukazatele na struktury `SXLOSILOJ` - jedna pro vyhledávání hesel, které začínají znaky'', ','. (`sercxsxlosiloj`), druhá pro ty, které zmíněnými znaky končí. Třetí ukazatel `SXLOSILOJ *lastasercxsxlosiloj` slouží pouze k upamatování, který ze dvou předchozích je aktuálně pro vyhledávání použit - tzn.který bude použit pro vyhledávání sousedů již nalezeného. Pro toto vyhledávání sousedů slouží i ukazovátko `montrilo`, ve kterém je pořadí právě nalezeného klíče v příslušné struktuře `SXLOSILOJ`. Poslední část struktury `TRADUKADO`, je `LISTO *sercxitaj`. Jedná se o oboustranný seznam, do kterého jsou ukládána nalezená hesla v případě, kdy za sousedy nalezeného hesla nelze považovat sousedy v pořadí `sercxsxlosiloj`, ani v pořadí `resercxsxlosiloj`. K tomu dojde tehdy, když na začátku ani na konci vyhledávaného hesla uživatel nezadá jeden ze znaků '', ','. Při tomto vyhledávání pak nelze využít primárně seřazených struktur `SERCXSXLOSILOJ`, a musí se porovnávat jedno heslo za druhým - viz.funkce `lantaTraduko`.

- **TRADUKADO* inicTradukado(char *aDATnomo)** - vytvoří a vrátí zinicializovanou strukturu `TRADUKADO`. V případě, že `aDATnomo` je `NULL`, je za datový soubor považován ten, který jehož umístění je uloženo v `defaultDATsituo`. Na začátku datového souboru je uloženo číslo `long int`, o které je třeba se pousunout, abychom se dostali k uloženým strukturám `SXLOSILOJ`. K načtení těchto struktur slouží již výše popsané metody `surekranigiSxlosiloj`.

- **FAKTO* leguFakto(FILE *f)** - předpokládá, že je soubor `*f` otevřen a čtecí pozice je na počátku sedmi textových řetězců[28], které se mají přečíst. Každý z těchto řetězců byl uložen pomocí makra `cpString` v souboru `malloko.h` - takže se nejdříve načte velikost řetězce do proměnné `l`, následně se podle této velikosti se alokuje paměť a čte příslušný počet znaků. Načtené řetězce jsou vráceny jako ukazatel `*FAKTO`.

- **FAKTO* traduku(TRADUKADO *t, char *vorto)** - podle zinicializované struktury `TRADUKADO` nalezne heslo `vorto` a vrátí externí data s tímto heslem související. Nejprve se zjišťuje, jestli na začátku, nebo na konci hledaného slova není jeden ze znaků '', ','. Pokud ano, je do ukazatele `lastasercxsxlosiloj` nastaven správný z obou klíčů a metodou 'dělení intervalů' je heslo vyhledáno. Pokud ne, pak nelze využít primárního seřazení ani v jednom z nich a je potřeba

[27] rozuměj struktury `TRADUKADO` a funkcí s ní zpřízněných

[28] viz. ENUMFAKTO ve 'vortaro.h': vortradiko, esperante, fonto, fako, cxehxe, noto, komparu

porovnávat hesla - jedno za druhým, o což se stará funkce `lantaTraduko`. Funkce `preparuDATdosiero`, zajišťuje nastavení čtecí pozice v dat-souboru a funkce leguFakto vrátí příslušná data[29].

- **void lantaTraduko(TRADUKADO *t, char *vorto)** - prochází jeden klíč za druhým a v případě shody uloží offset pro vyhledání externích dat, do seznamu `LISTO *sercxitaj`, který je součástí struktury `TRADUKADO`. Vnitřní ukazovátko tohoto seznamu je před opuštěním funkce nastaveno na první uložený prvek, protože právě ten má být jako první zobrazen.

- **void impresuApudulojn(TRADUKADO *t, char *vorto)** - vytiskne na konsoli všechny sousedy nalezeného hesla, která ještě vyhovují zadanému požadavku. Pokud bylo heslo nalezeno pomocí `lantaTraduko`, bude se procházet pouze seznam `LISTO *sercxitaj` a pro každou položku se:

  ⋆ **preparuDATdosiero, leguFakto** - načtou externí data související s položkou, na kterou ukazuje interní ukazovátko seznamu `LIST`.
  ⋆ **impresuFakton** - vytiskne načtená externí data[30]
  ⋆ **neniiguFakton** - uvolní načtená data z paměti
  ⋆ **sekvaListo** - posun interního ukazovátka seznamu `LIST` na další položku

  Pokud šlo pro vyhledání hesla využít jeden ze dvou primárně seřazených klíčů, provede se následující:

  ⋆ `sercxataVorto` - při vyhledávání podle `reasercxsxlosiloj` je potřeba vyhledávané heslo také obrátit (`reiguVorton`), v opačném případě se pouze okopíruje řetězec `vorto` (viz.`cpString`)
  ⋆ přeskočí se všechny klíče, které ukazují na stejná externí data[31]
  ⋆ pokud aktuální heslo ještě vyhovuje řetězci `sercxataVorto`, vytiskne se
  ⋆ poslední dva body se opakují, dokud heslo vyhovuje a nedošlo se do konce

  Funkce `impresuApudulojn` se používá pouze v neinteraktivním režimu, kdy je potřeba jednorázově dohledat další vyhovující položky.

- **void tradukadoInterakte (TRADUKADO *t)** - zajišťuje základní funkci programu v interaktivním režimu. V cyklu se čte vstup od uživatele a podle stisknutých kláves 'n', 'h', 'l' dochází k vyhledání zadaného hesla[32], nebo k posunu

[29] tedy překlad daného hesla
[30] tedy překlad hesla
[31] viz.prázdný cyklus `while`
[32] viz.funkce `traduku`

na sousední hesla (`irulefte/irudekstre`). Ukončení cyklu lze provést stiskem `'\021', '\006'` (ctrl+Q/F).

- **void tradukadoNoInterakte (TRADUKADO *t, char *vorto)** - nejvyšší funkce, která zajišťuje funkci překladu v neinteraktivním režimu:
  - ⋆ **impresuTitolon** - vytiskne výrazněji zadané heslo
  - ⋆ **traduku** - překlad hesla
  - ⋆ **impresuFakton** - výtisk překladu
  - ⋆ **neniiguFakton** - uvolnění překladu z haldy
  - ⋆ **impresuApudulojn** - vytiskne vyhovující okolí - viz.popis výše.

2.2.6 MemArangxo - kontrolní systém uvolnění všech alokovaných bloků v haldě

Pro spuštění kontrolního systému musí být splněny následující podmínky:

- **Linux** - program musí být spuštěn pod operačním systémem, který poskytuje prostřednictvím standartní knihovny `malloc.h` strukturu `mallinfo()`, pomocí které při každé (de)alokaci zjišťuji velikost alokované/uvolněné paměti. Na platformě Win by bylo potřeba použít poněkud jiné prostředky, zdá se mi nicméně zbytečné konstruovat kontrolní mechanizmus dublovaně.

- **DEBUG** - musí být definována tato symbolická konstanta. V hlavičkovém souboru `malloko.h` je potřeba přepsat konstantu `NDEBUG` na `DEBUG`. Pokud zůstává definováno `NDEBUG`, jsou všechny funkce `miaMalloc, miaFree` nahrazeny za standartní `malloc, free` a funkce `inicMemArangxo, finMemArangxo` jsou nahrazeny středníkem - viz.`malloko.h`.

- **erarsercxilo.txt** - v adresáři, kde se nachází program, lze tento soubor otevřít pro zápis. Sem bude program zapisovat při každé (de)alokaci velikost paměti. Na posledním řádku nám buď pogratuluje, nebo vyjádří soustrast pokud zůstane nějaká část paměti neuvolněna.

  Základní funkce:

- **void inicMemArangxo()** - voláno na začátku programu, založí soubor `erarsercxilo.txt` a vynuluje globální proměnnou `sumo`, která představuje celkovou velikost alokované paměti.

- **void* miaMalloc(char *noto, size_t dimensio)/miaFree()** - volá se místo standartního `malloc/free`. Rozdílem složek `uordblks` ze struktury `mallinfo` před a po volání standartního `malloc` se získá velikost (de)alokovaného prostoru.

Tato velikost je přičtena/odečtena k/od proměnné `sumo` a spolu s poznámkou `noto` zapsána do souboru `erarsercxilo.txt`.

- **void finMemArangxo()** - do souboru `erarsercxilo.txt` je zapsáno závěrečné zhodnocení podle toho, jestli je proměnná `sumo` nulová a soubor je uzavřen.

2.2.7 Symbolické konstanty a makra

- **NDEBUG**[33] - pokud je definována, nebude fungovat kontrolní systém `MemArangxo` a program se také vyhne několika kontrolním výpisům. Opak dosáhnete přepsáním na `DEBUG`.
- **miaFopen(variablo,nomo,parametro)**[33] - volání standartní funkce `variablo=fopen(nomo,parametro)`, pokud se nepovede soubor otevřít, ukončí program se slušným chybovým hlášením.
- **miaFclose(variablo)**[33] - volání standartní funkce `fclose(variablo)`, pokud se nepovede soubor zavřít, ukončí program se slušným chybovým hlášením.
- **cpString(t,f)**[33] - makro, které alokuje na ukazatel `t` tak velké množství paměti, aby do něj šlo překopírovat text, na který ukazuje `f` a samozřejmě také tuto kopii provede.
- **LINLONGO**[33]- nastaveno možná zbytečně opatrně na 1000, představuje maximální počet předpokládaných znaků na jeden řádek. Používá se k alokaci paměti při konverzi csv->dat[34]
- **NPARTF**[35]- počet položek v jednom záznamu[36] - ve slovníku, pro který je program `'vortaro'` vytvořen je tento počet 9.
- **VORTOLONGO**[35] - nejdelší výraz, jaký se předpokládá pro vyhledávání. Také možná trochu moc opatrně nastaveno na 100.
- **defaultDATnomo**[35] - nastaveno na 'espvortaro.dat', jde o název souboru, na který se bude program obracet o data, pokud nebude stanoveno přepínačem 'd' jinak.

---

[33] definována v `malloko.h`
[34] viz.ellaboruCSV, ceteraLinio
[35] definována v `vortaro.h`
[36] záznam alias FAKTO, viz.ENUMFAKTO ve `vortaro.h`

- **defaultDATsituo**[35] - absolutní cesta v adresářové struktuře k souboru defaultDATnomo, včetně samotného jména souboru. Tento textový řetězec je inicializován prostřednictvím funkce `trovuDATsituon`, která je volána z hlavní funkce `main`.

- **kusxiguTeksto(t,d)**[35] - slouží k uložení textového řetězce, na který ukazuje ukazatel `t` do souboru `FILE *d`. Nejprve se uloží délka řetězce[37] a pak samotný řetězec bez koncového '\0'.

- **leguTeksto(t,d)**[35] - pro ukazatel `t` alokuje paměť a přečte do ní text ze souboru `d` uložený pomocí `kusxiguTeksto`.

- **leguLONGINTpostSTR(str,li)**[35] - `str` je ukazatel na textový řetězec, za jehož ukončovacím znakem '\0' je uloženo číslo typu `long int`. Na toto číslo bude nasměrován ukazatel `li`.

- **NOMOENUMFAKTOn**[35] - zde jsou uloženy názvy jednotlivých částí v záznamu. Využívá se při tisku překladu - viz funkce `impresunomoenumfakto(int i)`.

- **ordo(i)**[35] - využívána ve funkci `impresuFakton` pro uspořádání výstupu. Díky tomuto makru se prohodí při tisku položka 'Fonto' s položkou 'Cxehxe'.

- **tdimensio/ndimensio**[38] - jsou konstanty, určující velikost paměti, kterou bude alokovat objekt[39] `SXLOSILOJ`. Více informací v kapitole 'SXLOSILOJ - klíče'

[37] pro t==NULL je délka také 0
[38] definována v `sxlosiloj.h`
[39] rozuměj struktury `SXLOSILOJ` a funkcí s ní zpřízněných

3 Rejstřík

%PATH% *6*, *8*
ĉeĥe *11*
ŝlosilo1 *11*
ŝlosilo2 *11*
_vortaro *5*
_vortaro.exe *8*

a
aDATnomo *20*
aldonuSxlosilo *14*, *15*, *16*
alLin *15*
alWin *15*
antauxaListo *19*
antauxaRingoListo *19*
auntauxaListo *19*

c
ceteraLinio *18*
cpString *20*, *21*
cpString(t,f) *23*
cxuMalplenaListo *19*

d
DATdosiero *20*
DEBUG *22*
defaultDATnomo *18*, *23*
defaultDATsituo *18*, *20*, *24*
Dekomprimace *4*, *6*, *7*
dekomprimace *9*
donuKonverto *18*, *19*

e
ELEMENT_LISTO *19*
ellaboruCSV *16*, *17*, *18*
erarsercxilo.txt *22*, *23*
esperante *11*
EspSlovnik.csv *5*, *7*, *9*, *10*, *15*
espvortaro.dat *5*, *7*, *12*

f
FAKTO *17*, *18*, *20*
fermuKonverto *19*
fermuSxlosiloj *16*
finMemArangxo *22*, *23*
fonto *11*
free *22*

i
impresuApudulojn *21*, *22*
impresuFakton *21*, *22*
impresuTitolon *22*
inicKodado *17*, *17*
inicKonverto *17*, *18*
inicListo *19*
inicMemArangxo *22*
inicSxlosiloj *16*
inicTradukado *20*
Instalace *4*, *6*, *8*
instalace *10*
instalurapide.win.bat *15*
instvort.lin *15*
instvort.win.bat *15*
integrace *3*
irudekstre *22*
irulefte *22*

k
kodado *14*, *17*
komparu *11*
Kompilace *4*, *8*
konservuSxlosiloj *14*, *16*, *18*
KONVERTO *2*
konverto *17*
KONVERTO *17*, *19*
konvertu *18*
konvertuKodado *17*
Konverze datového souboru *5*, *7*, *9*, *10*
KONVKODADO *2*, *16*

kusxiguTeksto(t,d) *24*

l
lantaTraduko *19*, *20*, *21*
lastasercxsxlosiloj *20*
leguFakto *20*, *21*
leguLONGINTpostSTR(str,li) *24*
leguMontriloListo *19*
leguTeksto(t,d) *24*
linio *18*
LINLONGO *23*
LIST *21*
LISTO *2*
listo *14*
LISTO *19*

m
makra *2*, *23*
malinstvort.lin *5*
malinstvort.win.bat *7*, *9*
mallinfo *22*
mallinfo() *14*, *22*
malloc *22*
malloc.h *22*
malloko *14*
malloko.h *20*, *22*
MemArangxo *2*, *22*
miaFclose(variablo) *23*
miaFopen(variablo,nomo,parametro) *23*
miaFree *22*
miaMalloc *22*

n
NDEBUG *22*, *23*
ndimensio *24*
ndimensio1 *16*
ndimensio2 *16*
neniiguFakton *21*, *22*
neniiguKodado *17*
neniiguListo *19*
nombrdimensio *16*
nombrmntr *16*
NOMOENUMFAKTOn *24*
noto *11*, *23*
NPARTF *17*, *23*
nulListo *19*
nulTutaListo *19*

o
OpenOffice *5*, *9*, *10*
ordo(i) *24*

p
partigiSxlosiloj *18*
Použití programu *2*, *11*
preparuDATdosiero *21*
purigu.lin *15*
purigu.win.bat *15*
přenositelnost *3*
přepínač '-k' *10*, *12*
přepínač '-kc' *9*, *10*
přepínač '-ki' *5*, *10*
přepínač '-ku' *5*
přepínač '-kw' *10*
přepínač '-t' *11*
přepínač '-t(i)d' *13*
přepínač '-ti' *12*
Případná odinstalace *5*, *7*, *9*

r
rangxiordigi *16*
rangxitaZono *16*
rangxu *16*
rangxuSxlosiloj *14*, *16*, *18*
reasercxsxlosiloj *17*, *18*, *21*
reiguVorton *17*, *21*
resercxsxlosiloj *20*
rychlost *3*

s
sekvaListo *19*, *21*
sekvaRingoListo *19*
sercxataVorto *21*

sercxitaj *20*, *21*
sercxsxlosiloj *17*, *20*
SERCXSXLOSILOJ *20*
setListo *19*
sumo *22*, *23*
surekranigiSxlosiloj *14*, *16*, *20*
sxlosilo1 *18*
sxlosilo2 *18*
SXLOSILOJ *2*
sxlosiloj *14*
SXLOSILOJ *14*, *15*, *16*, *17*, *18*, *20*
sxlosnombro *15*, *16*
sxlosteksto *15*, *16*
Symbolické konstanty *2*, *23*

t
tDATnomo *18*
tdimensio *24*
tdimensio1 *16*
tdimensio2 *16*
tekstdimensio *16*
tekstmntr *16*
TRADUKADO *2*, *20*, *21*
tradukadoInterakte *21*
tradukadoNoInterakte *22*
traduku *20*, *22*

u
uordblks *22*

v
Vortaro *5*, *6*, *7*, *7*, *9*, *9*, *10*
vortaro *17*
vortaro.c *14*, *16*, *19*
vortdosiero *5*, *12*
VORTOLONGO *23*
vortradiko *11*