Programování v jazyce C pro chemiky (C2160) – **závěrečné cvičení**

2. Kruhové schéma zobrazující propojení aminokyselin vodíkovou vazbou

Zadání

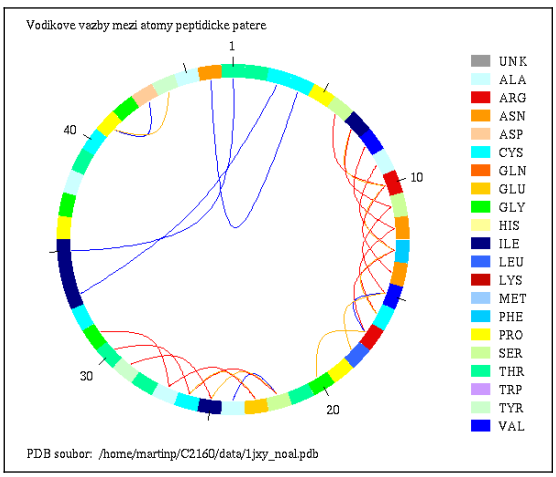
Vytvořte program, který určí aminokyseliny proteinu, jejichž atomy peptidické páteře jsou vzájemně spojeny vodíkovou vazbou. V kruhovém grafu budou tato residua spojena barevně rozlišenou křivkou. Program bude mít následující vlastnosti:

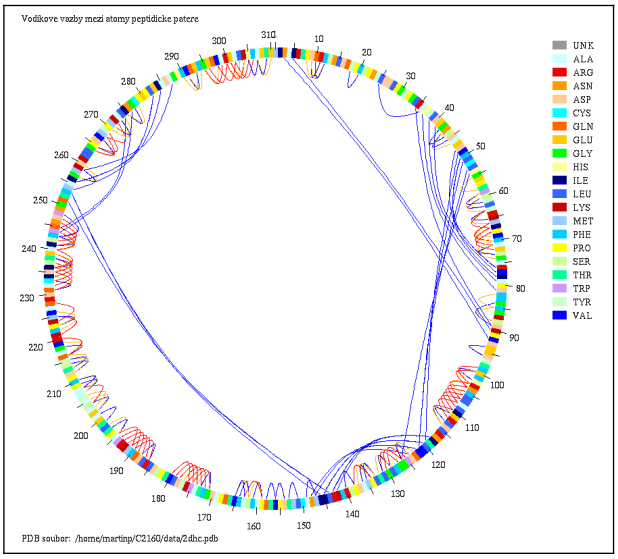
- Bude načítat PDB soubor s proteinem (pouze standardní residua z řádků ATOM)
- Určí atomy N a O peptidické páteře a na základě jejich vzdálenosti detekuje přítomnost vodíkové vazby mezi N jednoho residua a O druhého residua (vzdálenost do 3.2 A, nepočítá se pro sousedící residua)
- Bude vykresleno kruhové schéma, kde na okraji budou residua vyznačena barevně a také budou vyznačena čísla residuí (tj. pořadí v sekvenci jak je uvedeno v PDB souboru) – viz. obrázek níže
- Uvnitř grafu budou pomocí čar spojena residua mezi kterými existuje vodíková vazba. Barva čáry bude záviset na vzdálenosti residuí v sekvenci: červeně je-li vzdálenost 4 residua (typické pro alfa-helixy), oranžově pro 3 a fialově pro 5 residuí (typické pro nestandardní alfa-helixy), ostatní modře
- Graf bude obsahovat barevnou legendu, nadpis a jméno PDB souboru
- Jméno vstupního PDB souboru bude specifikováno jako parametr na příkazovém řádku
- Program bude uživatele informovat o chybě při otevření souboru, načítání konfiguračního souboru, překročení maximální přípustné velikosti polí a pod.
- Zdrojový kód programu bude opatřen komentáři

Nepovinné rozšíření (+5 bodů):

- Program bude načítat konfigurační soubor, ve kterém bude specifikováno jméno vstupního PDB souboru na řádku ve formátu "INPUT_FILE = jmeno_pdb_souboru", dále bude v konfiguračním souboru na samostatném řádku specifikována velikost okna ve formátu "WINDOW_SIZE = sirka, vyska"
- Název konfiguračního souboru bude předán programu jako parametr na příkazovém řádku

Program otestujte se strukturou crambinu (*1jxy_noal.pdb*) a enzymu haloalkan dehalogenáza (*2dhc.pdb*), které najdete mezi studijními materiály v IS MU ve složce „data“.





Dodržujte následující pravidla

- Dbejte na správné odsazování textu
- Pro reálné proměnné používejte typ `double`, ne `float`
- Při každém použití operátoru dělení si ujasněte, zdali dochází k celočíselnému nebo reálnému dělení a jaký typ dělení požadujete
- Proměnné vždy inicializujte vhodnou hodnotou
- Při použití funkcí pro práci s řetězci a při práci s poli dbejte na to, aby nedošlo k překročení velikosti pole
- Dobře zvažte, které proměnné budou lokální a které globální
- Názvy globálních proměnných volte tak, aby z nich byl jasný význam proměnné, volte raději delší názvy
- Názvy funkcí volte tak aby z nich bylo jasné jakou činnost funkce dělá
- Pro překlad programů používejte nástroj *make* (tj. vytvořte si příslušný *Makefile*)
- Z programu odstraňte veškerý kód který není nutný pro splnění zadání (např. pozůstatky z minulých cvičení, zakomentované části kódu). Ponechat můžete funkci pro zápis PDB
- Program nesmí při překladu vypisovat žádné varovné hlášky (při použití parametrů `-Wall -pedantic`)
- Na začátek programu umístěte stručný komentář obsahující jméno autora, rok vytvoření, popis funkce programu, parametry příkazového řádku, popř. formát konfiguračního souboru popisující činnost programu
- Všechny funkce a proměnné opatřete komentářem

Nápověda

1. Upravte funkci pro načítání PDB souboru tak, že bude načítat pouze řádky ATOM a nikoliv HETATM.
2. Ve struktuře proteinu vyhledejte pro každé residuum atomy peptidické páteře, tj atomy se jménem " N ", " CA ", " C ", " O " (vč. mezery na začátku a na konci) – viz. úloha 3 ze cvič. 9
3. Pro pohodlnější práci s těmito atomy, přidejte do struktury RESIDUE čtyři celočíselné proměnné (např. `atom_c, atom_c_alpha, atom_n, atom_o`) které budou obsahovat index příslušných atomů v poli atomů (tj. pořadí v poli atomů). Hodnoty proměnných nastavte pro každé residuum ve funkci pro vyhledávaní residuí nebo v samostatné funkci.
4. Vytvořte funkci, která vyledá atomy peptidické páteře vázané vodíkovými vazbami. Vodíková vazba vzniká mezi atomem peptidické páteře "O" a vodíkem na "N" jiného residua. Přesné určení vodíkové vazby vyžaduje znalost polohy H atomů (které často nejsou v PDB souborech obsaženy) a úhlů mezi atomy. Pokud stačí přibližné určení vodíkových vazeb, počítáme pouze vzdálenost mezi atomy "N" a "O", která by měla být < 3.2 Å. V programu nejdříve vytvořte strukturu `PEPTIDE_HBOND`, která bude obsahovat dvě proměnné (např. `residue1` a `residue2`), které budou indexem do pole residuí. Definujeme dostatečně velké pole `PEPTIDE_HBOND hbonds[]` a proměnnou `int hbonds_count`. Budeme počítat vzdálenosti atomů peptidické páteře, vždy každého atomu O a N residua s atomy O a N ostatních residuí (vždy porovnáváme jen N s O, ne O s O ani N s N), pokud je hodnota < 3.2 (tj. je mezi nimi vodíková vazba), uložíme záznam do pole `peptide_hbonds[]` (a zvětšíme hodnou `hbonds_count` o 1).
5. Kreslete kruh s barevnými kódy residuí podobně jako v uloze 1. ze cvičení 11. Je vhodné začít v horní části kruhu (tj. úhel 90° resp. π/2) a pokračovat po směru hodinových ručiček.
6. Do struktury RESIDUE přidejte proměnnou typu `double` (pojmenovanou např. `angle`) do které při každém vykreslení residua uložíte úhel odpovídající jeho pozici na kruhu. Toto bude dále použito pro kreslení spojovacích čar.
7. Vykreslete čáry spojující residua vázaná vodíkovou vazbou. Použijte cyklus přes všechny pole `peptide_hbonds[]`. Z hodnoty úhlů uložených v proměnné `angle` každého residua spočítejte počáteční a koncový bod čáry. V první fázi spojujte residua jen rovnými čarami.
8. Místo rovných čar propojte residua kubickou spline křivkou. K tomu použijte funkci `g2_raspln()` - viz. níže. Křivka bude dána třemi body, první a poslední bod jsou stejné jako při spojování rovnou čarou, bod pro vrchol oblouku zvolte v rozmezí úhlů krajních bodů a ve vzdálenosti uprostřed mezi středem a okrajem kruhu. Půlící úhel spočítáte jako rozdíl krajních úhlů dělený 2, pokud je však rozdíl mezi krajními úhly větší než 180°, musíte od výsledku odečíst 180°)
9. Dále je možné program upravit tak, aby bod pro vrchol oblouku závisel na počtu residuí v sekvenci mezi spojenými residui; čím více residuí, tím blíže ke středu kružnice bude vrchol oblouku (toto je nepovinné).

Funkce `g2_raspln()` kreslí kubickou spline křivku

```
void g2_raspln(int dev, int n, double * points, double tn)
```

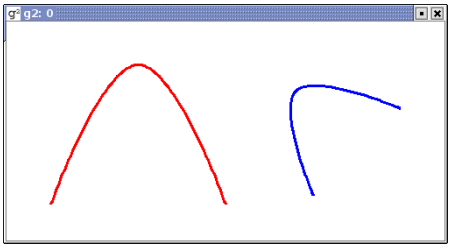
`n` je počet datových bodů, `points` je pole souřadnic `n` bodů `x1`, `y1`, ... `xn`, `yn`

`tn` je faktor zakřivení křivky (v rozsahu 0.0 až 2.0).

```
// Ukazka kresleni dvou kubickych spline
// krivek ktere je videt na obrazku dole
double points_a[6] = {50.0, 40.0, 150.0,
     200.0, 250.0, 40.0};
double points_b[6] = {350.0, 50.0, 330.0,
     170.0, 450.0, 150.0};

g2_set_line_width(dev, 3);
g2_pen(dev, 19);  // Cervena barva
// Kreslime kubickou spline krivku:
g2_raspln(dev, 3, points_a, 0.7);

g2_pen(dev, 3);  // Modra barva
g2_raspln(dev, 3, points_b, 0.7);
```



Testovací data

Seznam dvojic residuí (celkem 27), která jsou vázána vodíkovou vazbou pro strukturu crambinu (*1jxy_noal.pdb*):

```
THR1 - ILE35
CYS3 - ILE33
CYS4 - ASN46
SER6 - ARG10
ILE7 - ARG10
ILE7 - SER11
VAL8 - ASN12
ALA9 - PHE13
ARG10 - PHE13
ARG10 - ASN14
SER11 - VAL15
ASN12 - CYS16
PHE13 - CYS16
PHE13 - ARG17
```

```
VAL15 - ARG17
VAL15 - LEU18
ARG17 - GLY20
SER22 - ALA24
SER22 - ILE25
SER22 - CYS26
GLU23 - CYS26
GLU23 - ALA27
ILE25 - TYR29
CYS26 - THR30
ALA27 - GLY31
PRO41 - ASP43
PRO41 - TYR44
```