Úvod a zadání

Projekt BPPC	2012
Název projektu:	**CKontejner**
Autor 1:	`12656536`, **Richter** Miloslav, `richter@feec..`
Autor 2:	`23964754`, **Petyovský** Petr, `petyovsky@feec..`
Autor 3:	`ID`, **Příjmení** Jméno, `login/email`
Datum zadání:	16.10.2012
Datum odevzdání:	14.12.2012

Úvodní poznámky

V následujícím textu je vzor zadání. Tučně jsou označeny prvky, které se mohou lišit obsahem, ale které platí obecně pro třídu kontejner. Tyto pasáže je nutné nahradit upraveným významem pro konkrétní kontejner (normálním černým textem). Čím více tučných položek využijete, tím více vlastností si osvojíte a tím hodnotnější program bude. V tomto okamžiku nás nezajímá implementace, (tedy konkrétní realizace – názvy proměnných a jak uvnitř fungují metody), pouze u metod slovně popíšeme jak si představujeme jejich činnost. Poznámky v zadání nebudou, jsou pouze informativní. U některých tříd mohou nastat problémy s tím, že daný typ nemá pro některé metody nejlogičtější využití – zkuste si nějaké (byť méně logické) využití vymyslet (z hlediska programátorské praxe je to samozřejmě špatně, protože metody by měly být logické a snadno chápatelné, ale my zde dáme prioritu procvičení daného typu metod). Každý z autorů by měl napsat alespoň jednu metodu daného typu.

Vývoj projektu je možný v libovolném prostředí a kompilátoru C++, ale referenčním překladačem bude MSVC 2010.

Zadání

Popis datové struktury a výběr datových typů pro realizaci projektu

V rámci projektu se bude tvořit objektová knihovna zajišťující kontejnerové funkce pro prvky nesoucí hodnotu zvoleného typu. Vnitřní implementace kontejneru bude realizována jako obousměrně vázaný (lineární) kruhový seznam prvků. Prvky budou odvozené od dodané základní třídy **`CBNodeBase`**.

Zdrojové soubory základní třídy **`CBNodeBase`** budou dodány a je v rámci řešení projektu zakázána jejich modifikace.

Ze třídy **`CBNodeBase`** se odvodí třída `CBNode`, která bude prvkem kontejneru. Ve výsledném projektu bude vytvořený kontejner schopen pracovat minimálně se čtyřmi různými implementacemi třídy `CBNode`.

První dvě třídy `CBNode` Vám budou dodány v rámci zadání, společně se třídou **`CBNodeBase`** a realizují zapouzdření typu bool a výčtového typu `TWeekDay`. Podle těchto příkladů v rámci projektu vytvoříte dvě další třídy `CBNode`.

Kontejner budete realizovat bez podpory kontejnerových typů knihovny STL nebo jiných knihoven. Prvky kontejneru budou třídy odvozené z **`CBNodeBase`**.

Kontejner musí správně pracovat pro všechny čtyři vytvořené třídy `CBNode`, ale nemusí s nimi pracovat současně v rámci jednoho překladu.

První Vámi realizovaná třída prvku lineárního seznamu `CBNode` bude zapouzdřovat jeden ze základních typů, který si zvolte v části 1:

Zadání část 1: základní datové typy (mimo bool) (zvolte jednu z možností):

- Jeden z celočíselných typů: např.: (signed/unsigned) `char`, `short` int, `int`, `long` int
- Jeden z reálných typů: např.: `float`, `double`, `long double`
- Výčtový typ enum např.: (měsíce, výčty barev, znak Morseovy abecedy, znamení zvěrokruhu)

Druhá Vámi realizovaná třída prvku lineárního seznamu CBNode bude zapouzdřovat jeden ze složených datových typů, který si zvolte v části 2:

Zadání část 2: složené datové typy (struktury, pole) (zvolte jednu z možností):

- Komplexní číslo
- Textový řetězec (dynamický)
- Bod v prostoru (x,y,z)
- Číselný interval
- Barevná informace (RGB složky)
- Pole číselných hodnot typu double
- tón, krevní skupina, jméno a příjmení (dva členy),

Třetí Vámi realizovaná třída bude samotný kontejner pracující s prvky třídy CBNode. Charakter a funkce kontejneru si zvolte v části 3:

Zadání část 3: typ kontejneru implementován pomocí lineárního seznamu (zvolte jednu z možností):

- Zásobník – jeden vstup, jeden výstup, architektura LIFO
- Fronta – jeden vstup, jeden výstup, architektura FIFO
- Obousměrná fronta – jedna fronta, oba konce mohou sloužit vstup a výstup
- Prioritní fronta – prvky vstupují s informací o prioritě, prvky vystupují z fronty na základě priority a pořadí vstupu při stejné prioritě
- Množina – souhrn prvků, neumožňuje indexaci
- Multimnožina (množina obsahující i prvky stejné hodnoty) – tj. prvků, mohou se opakovat, …
- Mapa (asociativní pole) – umožňuje párování prvků CBNode a klíče celočíselné hodnoty (hash hodnoty)

Vzorový text zadání:

Todo:

Místo všech názvů **`CKontejner`** uveďte název kontejneru zvoleného v části zadání číslo 3

Tučně zapsané texty nahraďte modifikací pro vaše zadání (značky pro zvýraznění bold odstraňte). Nezvýrazněný text nemažte – jedná se o povinné body zadání.

Navrhněte třídu **`CKontejner`**, která bude realizovat knihovnu pro práci s **vybraným jednoduchým (`int`, `float` …)** typem, který bude zapouzdřen ve třídě `CBNode`, která bude potomkem dodané třídy **`CBNodeBase`**. Třída `CBNode` bude připravena pro práci s lineárním seznamem a její využití při tvorbě seznamu je povinnou součástí projektu. Tato knihovna bude realizovat činnost s **kontejnerem** podle následujících bodů (pokud je to možné, bude splňovat očekávané chování (podobné jako u jednoduchých typů (`int`, `float`...))). Navržená třída **CKontejner** umožní:

1. Třída `CBNode` zapouzdřuje datový typ: **jeden základní (`int`, `float`), jiný než `bool`**. Tuto definovanou třídu vytvoříme v jejím jmenném prostoru : **`CBNode_bool`, `CBNode_NAZEV_TYPU1`, `CBNode_NAZEV_TYPU2`.**

   Note:

   Zde uveďte za NAZEV_TYPU typy zvolené pro třídy z části zadání č.1 a č.2.

2. Vznik objektu – implicitní konstruktor **který vynuluje/naplní prvky na hodnoty …**, (konverzní)

konstruktor z `CBNode`, konstruktor **z typu int (vytvářející seznam o daném počtu prvků CBNode), …,** konstruktor z dvojice hodnot reprezentujících **pole prvků CBNode a počet prvků, …** , konstruktor na základě C řetězce (*char **), ve kterém budou hodnoty jednotlivých `CBNode` oddělené čárkami. Kopykonstruktor.

> **Note:**
> Celkem tedy – implicitní, kopykonstruktor, konverzní, konverzní z řetězce, s více parametry, … – (minimální počet 6ks)

3. Počet vzniklých a aktuálních instancí třídy **`CKontejner`** bude možné sledovat za pomoci statických proměnných třídy a bude možné je získat voláním statických metod. Každá instance třídy **`CKontejner`** bude obsahovat privátní proměnnou `iIndex` s unikátní hodnotou sloužící k identifikaci instance (odpovídající pořadí vzniku). Hodnota této proměnné musí být přístupná mimo třídu pomocí vhodné metody – **`VratIndex`**.
4. Zánik objektu – destruktor **kontejneru** bude implementován s mechanizmem odpočtu aktivních prvků přes statické proměnné. Dynamické členské prvky **ne/odalokuje**.
5. Budou implementovány operátory **kontejneru** s příslušnými činnostmi:
   - operátor = **pro přiřazení,**
   - operátor – **jako** unární operátor **pro inverzi kontejneru například jako: reverzace (otočení pořadí prvků), doplněk do plného kontejneru, nebo inverze jednotlivých hodnot CBNode.**
   - operátor – **jako** binární operátor **pro rozdíl kontejnerů,**
   - operátor + **pro** … **,**
   - operátor += **realizující a = a+b;**
   - logické operátory ==, !=, <=, > ,… **pro srovnání dvou kontejnerů (na základě obsahujících hodnot a jejich počtu).**
   - **libovolné další přetížitelné operátory**
   - **Dále** konverzní operátor **na `int`, který bude reprezentovat počet prvků kontejneru**.
   - Nečlenský operátor, využívající `friend` vlastností realizující **součet CBNode, `float`, `int` hodnoty a kontejneru**

   > **Note:**
   > Celkem tedy – unárních (nejméně 3ks), binárních (nejméně 3ks), konverzní, nečlenský, logické (nejméně 3ks) – (celkem minimálně 12ks).

6. Standardní vstup a výstup z instance třídy **`CKontejner`** bude realizován pomocí streamů (realizovaný `friend` funkcí s využitím jejích vlastností) – přetížením operátorů << a >> ve stejném formátu navrženém pro konstruktor z `char *` (tj.čárkami oddělený seznam hodnot).
7. Budou realizovány privátní a veřejné metody: veřejná metoda `PocetPrvku` pro zjištění prvků v **kontejneru**, která nebude mít parametry. Metody pro vložení a vyzvednutí prvku dle funkce **kontejneru. Dále metoda** … **která bude** …. Privátní metoda `Compare`, která porovná "velikost" dvou kontejnerů, kde velikost je dána počtem prvků, a vrátí hodnotu -1,0 nebo 1 (první je kratší, stejné, první je delší). Dále metoda `CompareDeep`, která porovná velikost kontejnerů i podle hodnot `CBNode`. Tyto metody budou používat logické operátory.

> **Note:**
> Pro privátní metody jsou nejvýhodnější funkce typu „uprav“, „přepočítej“, „zkontroluj“. Celkem tedy – minimálně 2ks privátní a 4ks veřejné.

8. Budou realizovány metody, pro reverzaci (**reverzace je otočení pořadí prvků kontejneru**):
   - umožňující volání `bbb=aaa.Reverzuj()`, která mění **kontejner** `aaa` na reverzovaný **kontejner** `bbb`,
   - umožňující volání `bbb=Reverzuj(aaa)`, která nechává **kontejner** `aaa` nezměněn a vrací (temp objekt) reverzovaný **kontejner**.
9. Použití dynamických dat ve třídě – **ne**předpokládám.
10. Dědění ve třídě – **ne**předpokládám.
11. Použití výjimek – **ne**předpokládám.
12. Ve vlastním projektu se předpokládá i implementace metod vytvářených překladačem implicitně.
13. Ve třídě bude pro kontrolu korektní práce s pamětí implementována knihovna `Check`.

Poznámky k řešení (jsou součástí zadání):

Odvozené třídy `CBNode_NAZEV_TYPU` vypracujte v projektu, kde je ve funkci `main` demonstrována činnost **`CBNode_bool`**. Po odladění je použijte ve vlastním projektu **kontejneru**, kde vytvořte v `main` podobný testovací kód (pro třídu kontejneru) jako je ukázáno v projektu s `CBNode`.

Vytvořená implementace **kontejneru** musí umožňovat bezchybný překlad s libovolnou třídou `CBNode` (zapouzdřující některý z jednoduchých datových typů) bez nutnosti zásahu do zdrojových textů **kontejneru**!!!! Implementace kontejneru tedy nebude závislá na typu ukládaných hodnot, protože bude využívat pouze definované rozhraní třídy `CBNode` (společné pro všechny typy), tak jak je uvedeno v testovacím příkladu. **CKontejner** proto implementujte tak, aby nevyužíval ani Setter a Getter metody třídy `CBNode`, jejichž hlavička je na zapouzdřujícím typu závislá!!!

Vlastní realizace třídy pro **kontejner** bude rozdělena na hlavičkový a zdrojový soubor. Další zdrojový soubor bude reprezentovat program demonstrující vlastnosti a použití definované třídy, který bude realizován jako konzolová aplikace přeložitelná ve Visual C++ (prázdný projekt, přísnost (stupeň) překladu 3 na detekci chybových a varovných hlášení). Soubory budou obsahovat úvodní poznámku o svém názvu, řešitelích ...).

Tento demonstrační program bude napsán tak, aby při činnosti nevyžadoval zásahy obsluhy ani vstupní data z jiných zdrojů (s výjimkou operátoru pro standardní vstup). Demonstrační programu bude inicializovat proměnné (v něm definovanými daty nezadávanými uživatelem) a na jejich základě bude demonstrovat činnost třídy a dále zobrazovat data na konzolu, nebo ukládat výsledky do souboru. K odevzdávanému zadání nezapomeňte připsat úvod pro hlavičku (tj. jména řešitelů, název projektu, datum zadání ... budou součástí odevzdávaného souboru)

Pro lepší orientaci uvádíme krátkou motivaci k pojmu třída na adrese http://www.uamt.feec.vutbr.cz/~richter/vyuka/1213_ppc/bppc/cviceni/motivace_trida.html.

Poslední změna: 2012-10-12

Id:

Introduction.txt 1 2012-10-15 13:25:08Z petyovsky