Llistes amb accés i modificació per índex X18544_ca

En aquest exercici, heu d'implementar un programa que simula una estructura de dades que és una mena de barreja entre llistes d'enters i vectors d'enters. Per una banda, hem de poder afegir elements pel principi i pel final de la llista. Per l'altra, hem de poder accedir i modificar indexadament els elements de la llista.
Aquest és un exemple d'entrada del programa:

```
v.push_back( 5 );
cout<<v[ 0 ];
v.push_front( 8 );
cout<<v[ 0 ];
cout<<v[ 1 ];
v.push_back( 9 );
cout<<v[ 0 ];
cout<<v[ 1 ];
cout<<v[ 2 ];
v[ 1 ]= 3 ;
cout<<v[ 0 ];
cout<<v[ 1 ];
cout<<v[ 2 ];
```

La sortida del programa amb aquesta entrada hauria de ser:

```
5
8
5
8
5
9
8
3
9
```

Com veieu a l'exemple d'entrada anterior, hi han espais en blanc envoltant cada número per a facilitar la lectura de l'entrada. Podeu llegir i tractar les comandes així:

```
...
int main()
{
    ...
string command;
while (cin >> command) {
    if (command == "v.push_back(") {
        int number;
        cin >> number;
        string ending;
```

```
        cin >> ending; // Això consumeix el ");"
        ...
    } else if (command == ...
    ...
}
}
```

Us recomanem que comenceu implementant una solució senzilla que superi els jocs de proves públics, obtenint així la meitat de la nota, i que mireu d'optimitzar-la més tard, si teniu temps.
Podeu utilitzar qualsevol de les estructures de dades presentades al curs (`vector`, `stack`, `queue`, `list`, `set`, `map`), i de la forma que considereu oportuna. Fixeu-vos, però, que enfocaments diferents donaran lloc a programes que seran més eficients o menys eficients, i d'això dependrà que pogueu superar només els jocs de proves públics o tots els jocs de proves, cosa que afectarà a la nota.
Existeixen enfocaments que aconsegueixen que totes les operacions tinguin cost constant. Tot i així, de cara a superar tots els jocs de proves, n'hi ha prou amb que aconseguiu que totes les operacions tinguin cost logarítmic com a molt.

Entrada

L'entrada del programa és una seqüència de línies, a on cada línia conté una comanda que pot ser d'un dels següents tipus:

```
v.push_front( NUMBER );
v.push_back( NUMBER );
cout<<v[ INDEX ];
v[ INDEX ]= NUMBER ;
```

A on `INDEX` és un natural entre 0 i el nombre d'elements afegits fins al moment menys 1, i `NUMBER` és un enter qualsevol.

Sortida

Per a cada instrucció `cout<<v[ INDEX ];` el programa escriurà el que suposadament conté la llista a la posició indexada per `INDEX` en aquell moment.

Exemple d'entrada 1

```
v.push_back( 5 );
cout<<v[ 0 ];
v.push_front( 8 );
cout<<v[ 0 ];
cout<<v[ 1 ];
v.push_back( 9 );
cout<<v[ 0 ];
cout<<v[ 1 ];
cout<<v[ 2 ];
v[ 1 ]= 3 ;
cout<<v[ 0 ];
cout<<v[ 1 ];
cout<<v[ 2 ];
```

Exemple de sortida 1

```
5
8
5
8
5
9
8
3
9
```

Exemple d'entrada 2

```
v.push_front( 0 );
cout<<v[ 0 ];
v[ 0 ]= 9 ;
v.push_back( -6 );
v[ 0 ]= 5 ;
v[ 0 ]= 7 ;
v.push_back( 2 );
v.push_front( -5 );
v[ 3 ]= 9 ;
v.push_front( -10 );
cout<<v[ 4 ];
cout<<v[ 3 ];
v[ 1 ]= 6 ;
cout<<v[ 3 ];
cout<<v[ 4 ];
v.push_front( 0 );
v.push_front( -8 );
v.push_back( 9 );
v.push_back( -2 );
v.push_front( 1 );
v[ 5 ]= -4 ;
v.push_front( 2 );
v.push_back( 9 );
v[ 5 ]= 2 ;
cout<<v[ 6 ];
cout<<v[ 2 ];
cout<<v[ 2 ];
v.push_front( -1 );
v[ 10 ]= 2 ;
v.push_front( 1 );
cout<<v[ 9 ];
v.push_front( -9 );
cout<<v[ 11 ];
cout<<v[ 4 ];
v[ 1 ]= -4 ;
cout<<v[ 14 ];
cout<<v[ 10 ];
v.push_back( -3 );
cout<<v[ 1 ];
v.push_front( -7 );
v[ 0 ]= -6 ;
v.push_back( 10 );
cout<<v[ 5 ];
v[ 5 ]= 9 ;
cout<<v[ 9 ];
v[ 15 ]= 1 ;
cout<<v[ 11 ];
cout<<v[ 13 ];
cout<<v[ 16 ];
v.push_front( 2 );
v.push_back( -9 );
v.push_front( -5 );
cout<<v[ 11 ];
v.push_back( -7 );
cout<<v[ 16 ];
v[ 7 ]= 0 ;
cout<<v[ 18 ];
v.push_back( 1 );
v.push_back( 2 );
v.push_front( 1 );
v.push_back( 4 );
v.push_front( -3 );
v.push_front( 6 );
v[ 3 ]= -2 ;
cout<<v[ 7 ];
v.push_back( -9 );
v[ 20 ]= 1 ;
v.push_back( 3 );
v.push_back( 2 );
v.push_back( 6 );
v[ 11 ]= 6 ;
v.push_back( -5 );
cout<<v[ 15 ];
cout<<v[ 20 ];
v[ 10 ]= 6 ;
v.push_front( -8 );
v[ 8 ]= 6 ;
cout<<v[ 32 ];
cout<<v[ 27 ];
v.push_front( -4 );
cout<<v[ 20 ];
v.push_back( 4 );
v[ 29 ]= 3 ;
cout<<v[ 29 ];
v.push_front( -2 );
cout<<v[ 19 ];
cout<<v[ 8 ];
v[ 9 ]= -10 ;
v[ 34 ]= -8 ;
v.push_back( -1 );
cout<<v[ 12 ];
v.push_front( 9 );
v.push_back( -6 );
v[ 21 ]= -1 ;
v[ 25 ]= 5 ;
v.push_front( 6 );
```

Exemple de sortida 2

```
0
9
-6
-6
9
-4
-8
-8
-6
9
1
9
-6
-4
1
```

```
2
-6
2
-3
2
-2
-3
-4
-4
1
6
2
2
3
-6
-6
2
```

Observació

Avaluació sobre 10 punts:

- Solució lenta: 5 punts.
- solució ràpida: 10 punts.

Entenem com a solució ràpida una que és correcta, de cost $O(n \log(n))$ i capaç de superar els jocs de proves públics i privats. Entenem com a solució lenta una que no és ràpida, però és correcta i capaç de superar els jocs de proves públics.

Informació del problema

Autor : PRO2
Generació : 2023-10-05 23:31:05