錬成問題

▪ メンバ関数でないにもかかわらず、クラスの非公開メンバにアクセスする特権が与えられるのが、 (1) 関数である。

▪ 単一の実引数で呼び出せるコンストラクタを、 (2) コンストラクタと呼ぶ。
(3) 関数を多重定義すると、クラス型のオブジェクトの値を任意の型へと変換できるようになる。一般に、Type 型に変換するための (3) 関数の名前は (4) である。変換関数の呼出しは、 (5) 。コンストラクタによる変換と、 (3) 関数による変換の総称が、 (6) 変換である。

▶ (5) の選択肢：
(a)必ずキャスト演算子によって行う
(b)必要に応じて暗黙裏に呼び出されるが、キャストによっても行える

▪ 演算子関数を多重定義すると、クラス型オブジェクトに対して演算子を適用できるようになる。一般に、演算子 op の関数名は (7) である。

▪ 増分演算子 `++` と減分演算子 `--` は、前置版と後置版の二つを区別して定義する。このうち、 (8) 版は (9) 型の引数を受け取る。

▪ ある演算子 `@` 用の演算子関数を定義すると、それに対応する複合代入演算子 `@=` 用の演算子関数がコンパイラによって自動的に定義されること (10) 。

▶ 選択肢：(a)はない　(b)になる

▪ 多重定義された論理演算子 `&&` と `||` による演算では、短絡評価が (11) 。

▶ 選択肢：(a)行われる　(b)行われない

▪ 関数の引数としてクラス型のオブジェクトを値渡しすると、そのオブジェクトのコピーが (12) コンストラクタによって作られる。なお、参照渡しでは、そのオブジェクトのコピーが (13) 。

▶ (13) の選択肢：
(a)デフォルトコンストラクタによって作られる
(b)作られない

▪ 異なる型のオブジェクトへの参照と、定数への参照は、`const` 参照でなければならない。その参照の参照先は、自動的に生成される (14) オブジェクトである。

▪ ヘッダ内のクラス定義の外で定義する非メンバ関数には、 (15) あるいは (16) のキーワードを付けて定義することによって、明示的に内部結合を与えなければならない。

▪ クラス型のオブジェクトを引数として受渡しを行う場合、 (17) 渡しでやりとりするのが原則である。

▪ 関数をフレンド関数とするためには、関数の冒頭に (18) を付けて定義しなければならない。

▪ 以下に示すクラス *MiniInt* は、0 以上 99 以下の整数を扱うクラスである。

```
class MiniInt {
    int v;          // 値
public:
    // コンストラクタ
    MiniInt(int value = 0) : v(value > 99 ? 99 : value < 0 ? 0 : value) { }
};
```

▫ このコンストラクタは、デフォルトコンストラクタとして機能 (19) 。また、変換コンストラクタとして機能 (20) 。また、コピーコンストラクタとして機能 (21) 。

▶ 共通の選択肢：(a)できる（する）　(b)できない（しない）

▫ クラス *MiniInt* に対して、以下の関数を追加して完全なクラスとせよ。なお、クラスはヘッダのみで実現するものとする（インクルードガードを施すこと）。加算や減算などの演算結果が 99 を上回る場合は 99 に調整し、0 を下回る場合は 0 に調整すること。

- **int** への変換関数（*v* の値をそのまま返却）
- 論理否定演算子 **!**（*v* が 0 であれば **true**、そうでなければ **false** を返却）
- 前置および後置の増分演算子 **++**
- 前置および後置の減分演算子 **--**
- 加算演算子 **+**（*MiniInt* **+ int**、**int +** *MiniInt*、*MiniInt* **+** *MiniInt* の演算に対応）
- 減算演算子 **-**（*MiniInt* **- int**、**int -** *MiniInt*、*MiniInt* **-** *MiniInt* の演算に対応）
- 複合代入演算子 **+=**（*MiniInt* **+= int**、*MiniInt* **+=** *MiniInt* の演算に対応）
- 複合代入演算子 **-=**（*MiniInt* **-= int**、*MiniInt* **-=** *MiniInt* の演算に対応）
- 等価演算子 **==, !=**（*MiniInt* **== int**、**int ==** *MiniInt*、*MiniInt* **==** *MiniInt* に対応）
- 関係演算子 **<, >, <=, >=**（等価演算子と同じ型のオペランドに対応）
- 挿入子 **<<**
- 抽出子 **>>**

このクラスを利用するプログラムも作成すること。… (22)