w8_1.cpp

```cpp
#include <iostream>
#include <cstdlib>
#include <cctype> // Included to use tolower() function

using namespace std;
class Month {
    public:
        /* We are using two different constructors here */
        Month(int m);
        Month();
        /* Member functions that uses input stream to get user data */
        void inputMonthNumber(istream& fin);
        void inputMonthChar(istream& fin);

        /* Member function to show to the user the information */
        void getMonthNumber(ostream& fout);
        void getMonthChar(ostream& fout);

        /* Here we create a new Month object and return to the user */
        Month getNextMonth();
    private:
        int month;
};

Month::Month(int m) {
    /* We need to make sure that the user input is valid
     * otherwise we simply return error */
    if (m > 12 || m < 1) {
        cout << "Invalid month!\n";
        exit(0);
    }
    else
        /* this->month refer to the member "month" of
         * the month class, so we use the "this->" to
         * refer to that. In most cases it is not necessary,
         * but because here we used same names, we need
         * to explicitly distinguish them. */
        month = m;
}
Month::Month () {
    cout << "!=====> Default constructor <=====!" << endl;
 /* Default constructor */
}
void Month::inputMonthNumber(istream& fin) {
    fin >> month;
    if (month > 12 || month < 1) {
        cout << "Invalid month!\n";
        exit(0);
    }
}
void Month::inputMonthChar(istream& fin) {
    char first, second, third;
```

```
53      fin >> first >> second >> third;
54
55      /* Here we need to convert (if needed) the user input
56       * in case the user used upper case letters */
57      first = tolower(first);
58      second = tolower(second);
59      third = tolower(third);
60
61      /* Now we convert the user input to the month value */
62      if (first == 'j' && second == 'a' && third == 'n')
63          month = 1;
64      else if (first == 'f' && second == 'e' && third == 'b')
65          month = 2;
66      else if (first == 'm' && second == 'a' && third == 'r')
67          month = 3;
68      else if (first == 'a' && second == 'p' && third == 'r')
69          month = 4;
70      else if (first == 'm' && second == 'a' && third == 'y')
71          month = 5;
72      else if (first == 'j' && second == 'u' && third == 'n')
73          month = 6;
74      else if (first == 'j' && second == 'u' && third == 'l')
75          month = 7;
76      else if (first == 'a' && second == 'u' && third == 'g')
77          month = 8;
78      else if (first == 's' && second == 'e' && third == 't')
79          month = 9;
80      else if (first == 'o' && second == 'c' && third == 't')
81          month = 10;
82      else if (first == 'n' && second == 'o' && third == 'v')
83          month = 11;
84      else if (first == 'd' && second == 'e' && third == 'c')
85          month = 12;
86      else {
87          cout << "Invalid month!\n";
88          exit(0);
89      }
90  }
91  Month Month::getNextMonth() {
92      /* First we need to see what is the currently month
93       * value. In case we are in month 12, we need to
94       * go back to month 1. */
95      int val = month < 12 ? (month+1) : 1;
96
97      /* Here we create a new Month object and return
98       * to the user. This new object is pointing to the
99       * next month. */
100     return Month(val);
101 }
102 void Month::getMonthNumber(ostream& fout) {
103     fout << month << endl;
104 }
105 void Month::getMonthChar(ostream& fout) {
106     /* We convert the month number to the
```

```
107         * respective characters. */
108        if (month == 1)
109            fout << "Jan";
110        else if (month == 2)
111            fout << "Feb";
112        else if (month == 3)
113            fout << "Mar";
114        else if (month == 4)
115            fout << "Apr";
116        else if (month == 5)
117            fout << "May";
118        else if (month == 6)
119            fout << "Jun";
120        else if (month == 7)
121            fout << "Jul";
122        else if (month == 8)
123            fout << "Aug";
124        else if (month == 9)
125            fout << "Sep";
126        else if (month == 10)
127            fout << "Oct";
128        else if (month == 11)
129            fout << "Nov";
130        else if (month == 12)
131            fout << "Dec";
132        fout << endl;
133    }
134    int main() {
135        /*************************************/
136        /* THIS PART IS THE SOLUTION FOR P1 */
137        /*************************************/
138
139        Month test1;
140        cout << "Enter the month by number: ";
141        test1.inputMonthNumber(cin);
142
143        cout << "Current month in number is: ";
144        test1.getMonthNumber(cout);
145        cout << "Current month in letter is: ";
146        test1.getMonthChar(cout);
147        cout << "Next month in number is: ";
148
149        Month a = test1.getNextMonth();
150        a.getMonthNumber(cout);
151        cout << "Next month in letter is: ";
152        a.getMonthChar(cout);
153
154        /*************************************/
155        /* THIS PART IS THE SOLUTION FOR P2 */
156        /*************************************/
157
158        Month test2;
159        cout << "Enter the month by letters: ";
160        test2.inputMonthChar(cin);
```

```
161
162     cout << "Current month in number is: ";
163     test2.getMonthNumber(cout);
164     cout << "Current month in letter is: ";
165     test2.getMonthChar(cout);
166     cout << "Next month in number is: ";
167     Month b = test2.getNextMonth();
168     b.getMonthNumber(cout);
169     cout << "Next month in letter is: ";
170     b.getMonthChar(cout);
171
172     /************************************/
173     /* THIS PART IS THE SOLUTION FOR P3 */
174     /************************************/
175
176     int number;
177     cout << "Enter the month by number: ";
178     cin >> number;
179     Month test3(number);
180     cout << "Current month in number is: ";
181     test3.getMonthNumber(cout);
182     cout << "Current month in letter is: ";
183     test3.getMonthChar(cout);
184     cout << "Next month in number is: ";
185     Month c = test3.getNextMonth();
186     c.getMonthNumber(cout);
187     cout << "Next month in letter is: ";
188     c.getMonthChar(cout);
189
190     cout << endl;
191     return 0;
192 }
```

w8_2.cpp

```cpp
#include <iostream>
#include <vector>
#include <stdio.h>
#include <string.h>

using namespace std;

class Todo {
    private:
        /* String stores the task description */
        string task;
    public:
        Todo();
        /* We pass the cin stream to read data from user */
        void setTask(istream& fin);
        /* We are returning the string reference that
         * actually stores the task description */
        const string&  getTask() const;
};

class TodoList {
    private:
        /* Here is the vector that stores Todo types */
        static vector<Todo> list;
        static int size;
    public:
        /* Default constructor */
        TodoList();
        /* We pass the Todo object by reference */
        void addTodo(Todo& t);
        void printList();
        static int getSize();
};

Todo::Todo() {
}

void Todo::setTask(istream& fin) {
    /* getline() is a built in function to read a line
     * from from the standard input.
     * the ws parameter is to extract the spaces that
     * user may type while describing the task. Thus
     * we can safely store all the description.
     * If we don't specify this, the any space will be
     * considered as the end of the input from the user */
    getline(fin >> ws, task);
}

/* We return a const string& reference, because we don't want
 * that the string can be modified outside once we return it */
const string& Todo::getTask() const {
    return task;
```

```
}

/* Static members must be initialized before used */
int TodoList::size = 0;
vector<Todo> TodoList::list;

void TodoList::printList() {
    for (int i = 0; i < size; i++) {
        cout << "[" << i+1 << "]: " << list[i].getTask() << endl;
    }
}

TodoList::TodoList() {}

void TodoList::addTodo(Todo& t) {
    /* Here we are inserting the Todo type to the vector */
    list.push_back(t);
    size++;
}

int TodoList::getSize() {
    return size;
}

int main() {
    TodoList list;
    int iter;
    cout << "How many tasks to be created? ";
    cin >> iter;
    Todo todo[iter];
    while(iter--) {
        cout << "Set task string: ";
        todo[iter].setTask(cin);
        list.addTodo(todo[iter]);
    }
    list.printList();
    cout << "List size = " << list.getSize() << endl;
}
```