Exercise 5.2a

```
#include <iostream> // for cout
using namespace std;
class Cat           // declare the class Cat
{
public:  void SetAge (int age);
      int GetAge();
      void SetWeight (int weight);
      int GetWeight();
private:  int itsAge;
      int itsWeight;
};

int Cat::GetAge()
{   return itsAge;
}

void Cat::SetAge(int age)
{   itsAge = age;
}

int Cat::GetWeight()
{   return itsWeight;
}

void Cat::SetWeight(int weight)
{   itsWeight = weight;
}

int main()
{
    int age, weight;
    Cat Felix;
    cout << "Felix is a cat who is "<< Felix.GetAge() << " years old and "<<
Felix.GetWeight() << " lb.\n";
    cout << "\nPlease enter Felix new Age and Weight: ";
    cin >> age >> weight;
    Felix.SetAge(age);
    Felix.SetWeight(weight);
    cout << "Felix is now " << Felix.GetAge() << " years old and " <<
Felix.GetWeight() << " lb.\n";
    return 0;
}
```

Ex. 5.2b

```
#include <iostream> // for cout
using namespace std;
class Cat // declare the class Cat
{
public:  Cat(); // Constructor
    ~Cat(); // Destructor
    void SetAge (int age);
    int GetAge() const;
    void SetWeight (int weight);
    int GetWeight() const;
private:  int itsAge;
    int itsWeight;
};
// Constructor
Cat::Cat()
{   cout << "Constructor......\n";
    itsAge = 5;
    itsWeight = 10;
}

// Destructor
Cat::~Cat()
{   cout << "Destructor.......\n";
}

void Cat::SetAge(int age)
{   itsAge = age;
}
```

```
int Cat::GetAge() const
{   return itsAge;
}

void Cat::SetWeight(int weight)
{   itsWeight = weight;
}

int Cat::GetWeight() const
{   return itsWeight;
}

int main()
{
    int age, weight;
    Cat Felix;
    cout << "Felix is a cat who is "<< Felix.GetAge() << " years old and "<<
Felix.GetWeight() << " lb.\n";
    cout << "\nPlease enter Felix new Age and Weight: ";
    cin >> age >> weight;
    Felix.SetAge(age);
    Felix.SetWeight(weight);
    cout << "Felix is now " << Felix.GetAge() << " years old and " <<
Felix.GetWeight() << " lb.\n";
    return 0;
}
```

Ex. 5.2c

```
#include <iostream>     // for cout
using namespace std;
class Cat       // begin declaration of the class
{
public:     // begin public section
    Cat(int initialAge);        // constructor
    ~Cat();         // destructor
    int GetAge(); //const;      //GetAge() does change the value of itsAge
    void SetAge(int Age);   // accessor function
    void Meow();            // general function
private:
    int itsAge;         // member variable
};

Cat::Cat(int initialAge)
{   itsAge = initialAge;
    cout << "Cat Constructor\n";
}

Cat::~Cat()
{   cout << "Cat Destructor\n";
}

int Cat::GetAge()
{   return (itsAge++); //Because itsAge changes, GetAge() must not be const
}

void Cat::SetAge(int age)
{   itsAge = age;
}

void Cat::Meow()
{   cout << "Meow.\n";
}

int main()
{   Cat Frisky(5);  //Constructor needs initialized value
    cout << Frisky.GetAge() << " years old\n";
    Frisky.Meow();
    //Frisky.Bark();  Bark() is not defined
    //Frisky.itsAge = 7;  itsAge is private
    Frisky.SetAge(7);
    cout << Frisky.GetAge() << " years old\n";
    return 0;
}
```