電子情報工学実験報告
実験 10　C プログラム演習　C++の基礎

報告者：4D–29　永安 佑希允
共同実験者：伊藤剛志，高橋秀和，馬場要一郎
指導教官：滝沢教官

実験日：1999 年 04 月 26 日，05 月 10 日
提出日：1999 年 05 月 17 日

1 目的

C++言語は，*Bjarne Stroustrup* によって C 言語をもとに拡張された言語であり C のスーパーセットである。C 言語に対して様々な改良が施されており，そのままベターな C として使うこともできるが，クラスの概念がもっとも重要な点である。この機能でクラスを定義したり，継承したりする事ができる。それによって，異なる抽象のレベルをそれぞれ独立してプログラムすることができる。

一昨年の 1998 年にようやく標準が策定され，Standard Template Library などを使って，C と比べると遙かに記述量の少ないプログラムを可能にしている。しかし，普及しながら様々な変更が加えられてきた経緯から，言語としてあまりにも巨大になってしまい，それに対する批判も少なくない。特に STL を使った場合，コンパイル時間は長くなる。

2 課題

2.1 オブジェクトとメッセージの関係

次のプログラムにおいて，

```
#include<stream.h>

class Format_J{
public:
  void outout(char* s, int d) const
    {cout << "私は " << s << " という名前で，年齢は " << d << " 歳です．" << endl;}
};

class Format_E{
public:
  void outout(char* s, int d) const
    {cout << "I am " << s << ", and " << d << " years old." << endl;}
};

main()
{
  Format_J *format_j = new Format_J;
  Format_E *format_e = new Format_E;

  format_j->output("Bill", 13);
  format_j->output("Michael", 24);
  format_j->output("Louis", 5);

  format_e->output("Bill", 13);
  format_e->output("Michael", 24);
  format_e->output("Louis", 5);

  delete format_j;
  delete format_e;
}
```

オブジェクトとしては，それぞれ `Format_J` と `Format_E` のインスタンスが存在する。それらのインスタンス即ちオブジェクトは，コンパイラの用意したメモリ管理システムが OS から与えられた領域に存在する。それらオブジェクト直接表現した変数はなく，`format_j` 及び `format_e` がそのインスタンスへのポインタ

として存在する。

それらのオブジェクトに対して，それぞれ 3 人ずつの名前と年齢を与えて，その情報をクラス定義で定義した方法，この場合は日本語と英語で std::cout に出力せよとのメッセージを送っている。

しかしながら，このやり方はあまりスマートではない。本来なら，純粋仮想関数 output を持つ抽象クラス Format を作り，Format_J と Format_E はそこから派生させるべきである。また output のメッセージをオブジェクトに送る部分も，Format*型の引数を持つ別に関数に代替させることにより，その関数は実際のオブジェクトの正確な型を知らなくてもオブジェクトに対して output させることができる。

しかしこの方法でも，同一のインターフェースがあることにより，クラス定義の違いを意識せずにオブジェクトを使うことができるという点で，オブジェクト指向なプログラムであると言える。

2.2 クラス Format_A の作成

問題のクラスを含むプログラムは，次のようになる。

```
#include<stream.h>

class Format_A{
public:
  void outout(char* s, int d) const {
    if(d >= 20)
      cout << "私は " << s << " という名前です．" << endl;
    else
      cout << "私は " << s << " という名前で，年齢は " << d << " 歳です．" << endl;
  }
};

main()
{
  Format_A *format_a = new Format_A;

  format_a->output("Bill", 13);
  format_a->output("Michael", 24);
  format_a->output("Louis", 5);

  delete format_a;
}
```

このプログラムは，純粋仮想関数によるインターフェース定義や多態の実現はできないが，同一のインターフェースがあることにより，クラス定義の違いを意識せずにオブジェクトを使用することができている。この点で，オブジェクト指向なプログラムであると言える。

3 考察

ここでは，標準 C++[注1]によるプログラムの例を載せておく。標準ライブラリの std::string の使用や，それに伴う名前空間の概念の導入などがある。また，抽象クラスを使った多態についても実現させてみた。

```
#include<iostream>
#include<string>
```

[注1] ISO/ISC 14882, Standard for C++ Programming Language

```
class Format{
public:
  virtual void outout(std::string name, int age) const = 0;
};

class Format_J : public Format{
public:
  void outout(std::string name, int age) const
    {std::cout << "私は " << name << " という名前で,年齢は " << age << " 歳です. " << endl;}
};

class Format_E : public Format{
public:
  void outout(std::string name, int age) const
    {std::cout << "I am " << name << ", and " << age << " years old." << endl;}
};

void out(Format *format)
{
  format->output("Bill", 13);
  format->output("Michael", 24);
  format->output("Louis", 5);
}

main()
{
  Format_J *format_j = new Format_J;
  Format_E *format_e = new Format_E;

  out(format_j);
  out(format_e);

  delete format_j;
  delete format_e;
}
```

また，一連のプログラムにわたって “Bill”, “Michael”, “Louis” という名前が出てくるが，出題者の傾向からして，コンピューターの世界での有名な人か，よく使われるコンピューターの名前であるように思えてならない。