Esercizi per il corso di Fondamenti di Informatica I

Tratti dal libro: "Introduzione alla programmazione ad oggetti in Java" – C.Thomas Wu – McGraw Hill – 2000

Esercizio 1

Si identifichino gli errori nel seguente programma

```
/*
      Program Esercizio 1
      Un programma con molti errori.
//
import javabook.mainwindow;

class Esercizio 1
{
      public void Main()
      {
            MainWindow mainWindow;
            mainWindow.show();
      }
}
```

Esercizio 2

Si identifichino tutti gli errori nel seguente programma

```
//
      Programma Esercizio 2
      Un programma con molti errori.
//

import JavaBook.*;

class TWO
{
      public static void main method()
      {
            mainWindow mainWindow;
            MessageBox mybox1, mybox2;

            mainWindow = new MainWindow();
            messageBox.show();
            mybox2 = new MessageBox();
            mybox2.show;
      }
}
```

Esercizio 3

Si descriva lo stato della memoria dopo aver eseguito ciascuna delle seguenti istruzioni:

```
MainWindow window1;
MessageBox mbox1, mbox2;

window1 = new MainWindow();
mbox1   = new MessageBox( window1 );
mbox2   = new MessageBox( window1 );
```

Esercizio 4

Si descriva lo stato della memoria dopo aver eseguito ciascuna delle seguenti istruzioni:

```
Person person1, person2;

person1 = new Person();
person2 = new Person();
person2 = new Person();
```

Esercizio 5

Si scriva un'applicazione Java che mostri una finestra `MainWindow` larga 300 pixel e alta 200 pixel, con il titolo `Il mio primo programma`. Le dimensioni di una finestra possono essere cambiate inviando il messaggio `setSize` alla finestra. La sintassi `setSize` è la seguente:

```
setSize( <width>, <height> )
```

Esercizio 6

Nell'esercizio precedente si è utilizzato il metodo `setSize` per impostare le dimensioni di una finestra `mainWindow`. Questo metodo consente di impostare la grandezza di una finestra, ma non la posizione sullo schermo della stessa. Per collocare una finestra in una posizione precisa, si utilizza il metodo `setLocation` come segue:

```
// mainWindow è stata dichiarata e creata

mainWindow.setLocation( 50, 50 );
mainWindow.setVisible( true );
```

Mediante alcuni esperimenti, si determini come il valore degli argomenti di `setLocation` determini la posizione della finestra.

Esercizio 7

Si scriva un'applicazione Java che visualizza due messaggi separati: "Posso progettare" e "Posso programmare".

Esercizio 8

Si scriva un'applicazione Java che visualizza un messaggio molto lungo. Si provi con un messaggio che eccede le dimensioni dello schermo del computer.

Esercizio 9

Data la velocità degli odierni calcolatori, i due programmi seguenti non risulteranno molto diversi nel loro comportamento

```
MainWindow myWindow;
myWindow = new MainWindow( );
myWindow.setVisible( true );
```

```
MainWindow myWindow;
myWindow = new MainWindow( );
myWindow.setVisible( true );
myWindow.setVisible( false );
myWindow.setVisible( true );
```

Un modo per rallentare il programma a destra in modo da rendere visibile il cambiamento, è di introdurre una pausa tra i messaggi setVisible. Per inserire una pausa, si può utilizzare un oggetto Clock del package javabook. Ad esempio:

```
Clock myClock;
myClock = new Clock( );

// istruzione X
myClock.pause( 2 );   // Introduce una pausa di 2 secondi
// istruzione Y
```

Utilizzando la classe Clock, si scriva un programma che faccia apparire un oggetto MainWindow per cinque secondi, lo renda invisibile per 3, e lo faccia nuovamente riapparire.

Esercizio 10

Si supponga di avere le seguenti dichiarazioni:

```
int i = 3, j = 4, k = 5;
float x = 34.5f, y = 12.25f;
```

Si determini il valore di ciascuna delle seguenti espressioni o si spieghi perchè non sono espressioni valide:

```
  a. ( x + 1.5) / (250.0 * (i/j));
```

```
b. x + 1.5 / 250.0 * i / j;
c. –x * -y * (i + j) / k;
d. (i  / 5) * y;
e. Math.min(i, Math.min(j, k));
f. Math.exp(3, 2);
g.  y % x;
h. Math.pow(3, 2);
i. (int) y % k;
j. i / 5 * y;
```

Esercizio 11

Si supponga di avere le seguenti dichiarazioni:

```
int m, n, i = 3, j = 4, k = 5;
float v, w, x = 34.5f, y = 12.25f;
```

Si determini il valore assegnato alla variabile in ciascuna delle seguenti istruzioni di assegnamento, o si spieghi perchè non è un assegnamento valido:

```
a. w = Math.pow(3, Math.pow(i, j));
b. v = x / i;
c. w = Math.ceil(y) % k;
d. n = (int) x / y * i / 2;
e. x = Math.sqrt( i*i – 4*j*k);
f. m = n + i * j;
g. n = k / (j * i) * x + y;
h. i = i + 1;
i. w = float( x + i );
j. x = x / i / y / j;
```

Esercizio 12

Si supponga di avere le seguenti dichiarazioni:

```
int i, j;
float x, y;
double u, v;
```

Quali dei seguenti assegnamenti sono validi?

```
a. i = x;
b. x = u + y;
c. x = 23.4 + j * y;
d. v = (int) x;
e. y = j / i * x;
```

Esercizio 13

Si scrivano le espressioni Java corrispondenti alle seguenti espressioni algebriche:

a. La radice quadrata di $B^2 + 4AC$ (A e C sono variabili distinte).
b. La radice quadrata di $X + 4Y^3$.
c. La radice cubica del prodotto di X e Y.
d. L'area di una circonferenza (πR^2).
e. Il seno di C fratto il seno di A.

Esercizio 14

Si determini cosa visualizza il seguente programma senza eseguirlo:

```
import javabook.*

class TestOutputBox
{
      public static void main ( String args[] )
      {
            MainWindow mainWindow;
            OutputBox outputBox;

            mainWindow = new MainWindow(“Programma TestOutputBox”);
            outputBox = new OutputBox(mainWindow);

            mainWindow.setVisible( true );
            outputBox.setVisible( true );

            outputBox.printLine(“Uno”);
            outputBox.print(“Two”);
            outputBox.skipLine(1);

            outputBox.print(“Tre”);
            outputBox.printLine(“Quattro”);
            outputBox.skipLine(1);

            outputBox.print(“Cinque”);
            outputBox.printLine(“Sei”);
      }
}
```

Esercizio 15

Si determini il risultato delle istruzioni seguenti:

```
int x, y;
x = 1;
y = 2;
messageBox.show(“Il risultato è “ + x + y);
messageBox.show(“Il risultato è “ + (x + y) );
```

Esercizio 16

Scrivere un programma che visualizzi il seguente disegno in un `OutputBox:`

```
0X0X0X0X0X0X0X0X0X0X
X                   0
0                   X
X                   0
0                   X
X                   0
0X0X0X0X0X0X0X0X0X0X
```

Esercizio 17

Scrivere un programma che converta centimetri in pollici. Si utilizzi `InputBox` per l'inserimento dei dati e `OutputBox` per la visualizzazione del risultato (1 pollice = 2.54 centimetri).

Esercizio 18

Scrivere un programma che converta temperature Celsius in temperature Fahrenheit. Si utilizzi `InputBox` per l'inserimento dei dati e `OutputBox` per la visualizzazione del risultato. La scala Fahrenheit è divisa in 180 gradi, e lo 0 Celsius corrisponde a 32 gradi Fahrenheit.

Esercizio 19

Scrivere un programma che richieda il peso di una persona e mostri il numero di calorie giornaliere che la persona necessita. Una persona necessita di circa 30 calorie giornaliere per chilogrammo di peso (Nota: non stiamo distinguendo tra maschi e femmine).

Esercizio 20

Una quantità nota come BMI (Body Mass Index, indice di massa corporea) viene utilizzata per calcolare la probabilità di problemi di salute associati al peso. La formula per il calcolo del BMI è la seguente:

$$BMI = \frac{w}{h^2}$$

dove w è il peso in chilogrammi e h è l'altezza in metri. Un BMI da 20 a 25 è considerato "normale". Si scriva un programma che dati peso e altezza calcola il BMI.

Esercizio 21

Si scriva un programma per trovare le soluzioni di un'equazione di secondo grado $aX^2+bX+C=0$, assumendo che $A \neq 0$ e che $b^2-4ac \geq 0$.

Esercizio 22

Un grossista di caffè utilizza sacchi da 1 chilogrammo per spedire il prodotto ai clienti. Ogni sacco costa 6 euro. Quando un cliente fa un ordine, la merce viene spedita imballando i sacchi in tre tipi di scatole: grande (20 sacchi), media (10 sacchi) e piccola (5 sacchi) del costo, rispettivamente, di 2, 1 e 0.5 euro. L'ordinativo viene inviato al cliente minimizzando il numero di cartoni. Per esempio un ordine di 25 sacchi verrà spedito usando un cartone grande e un cartone piccolo. Scrivere un programma che calcoli il costo totale dell'ordine. Si utilizzi un oggetto `InputBox` per richiedere il numero dei sacchi ordinati e un oggetto `OutputBox` per visualizzare:

- il costo totale del solo caffè
- il numero di cartoni utilizzato per ogni formato
- il costo totale dell'ordine

Esercizio 23

Si consideri la seguente classe istanziabile:

```
class Esercizio23
{
	public final int A = 345;
	public int b;
	private float c;

	private void metodoUno( int a )
	{
		b = a;
	}
	public float metodoDue( )
	{
		return 23;
	}
}
```

Si identifichino e si motivino le istruzioni non valide nella seguente classe principale:

```
class Principale
{
	public static void main( String[ ] args )
	{
		Esercizio23 e23;
		e23 = new Esercizio23( );

		e23.A = 12;
		e23.b = 12;
		e23.c = 12;
		e23.metodoUno( 12 );
		e23.metodoUno( );
		System.out.println( e23.metodoDue( 12 ) );
		e23.c = e23.metodoDue( );
	}
}
```

Esercizio 24

Qual è il risultato dell'esecuzione del seguente programma?

```
class Principale
{
	public static void main( String[ ] args )
	{
		Esercizio24 e24 = new Esercizio24( );
		e24.init( );

		e24.increment( );
		e24.increment( );
		System.out.println( e24.getCount( ) );
	}
}

class Esercizio24
{
	private int count;

	public void init( )
	{
		count = 1;
	}
	public void increment( )
	{
		count = count + 1;
	}
	public int getCount( )
	{
		return count;
	}
}
```

Esercizio 25

Qual è il risultato dell'esecuzione del seguente programma?

```
class Principale
{
	public static void main( String[ ] args )
	{
		Esercizio25 e25 = new Esercizio25( );
		e25.init( );

		e25.count = e25.increment( ) + e25.increment( );
		System.out.println( e25.increment( ) );
	}
}
```

```
class Esercizio25
{
	public int count;

	public void init( )
	{
		count = 1;
	}

	public int increment( )
	{
		count = count + 1;
		return count;
	}
}
```

Esercizio 26

Si determini il risultato dell'esecuzione del seguente programma.

```
import javabook.*;

class Esercizio26
{
	private int x, y, z;
	private MainWindow mainWindow;
	private OutputBox outputBox;

	public void start( )
	{
		int x, y;

		setup( );
		x = y = 10;
		modify(x, y);
		printout( );
	}

	private void setup( )
	{
		mainWindow = new MainWindow( );
		outputBox = new OutputBox(mainWindow);
		mainWindow.setVisible( true );
		outputBox.setVisible( true );

		x = 100;
		y = 200;
		z = 300;
	}
```

```
      private void modify( int x, int y )
      {
            z = x + y;
            x = z;
            y = 2 * z;
      }

      private void printout( )
      {
            outputBox.printLine(“x = “ + x);
            outputBox.printLine(“y = “ + y);
            outputBox.printLIne(“z = “ + z);
      }
}

class Principale
{
      public static void main( String[ ] args )
      {
            Esercizio26 e26 = new Esercizio26( );
            e26.start ( );
      }
}
```

Esercizio 27

Migliorare il seguente programma utilizzando due classi, di cui una istanziabile (`Esercizio27`) e un’altra contenente il programma principale (`main`).

```
class Esercizio27
{
      public static void main( String[ ] args )
      {
            double radius;
            double circumference;

            MainWindow mainWindow;
            OutputBox outputBox;
            InputBox inputBox;

            int smallRadius, largeRadius;
            double smallCircum, largeCircum;

            mainWindow = new MainWindow( );
            outputBox = new OutputBox(mainWindow);
            inputBox = new InputBox(mainWindow);

            // Calcolo della circonferenza per un cerchio piccolo
            smallRadius = inputBox.getDouble(“Raggio del cerchio
                                              piccolo? ”);
            radius = smallRadius;
            smallCircum = 2 * Math.PI * radius;
```

```
            // Calcolo della circonferenxa di un cerchio grande
            largeRadius = inputBox.getDouble("Raggio del cerchio
                                              grande? ");
            radius = largeRadius;
            largeCircum = 2 * Math.PI * radius;

            // Visualizza la differenza
            outputBox.printLine("Risultati:");
            outputBox.printLine("Circonferenza piccola: " +
                                smallCircum);
            outputBox.printLine("Circonferenza grande: " +
                                largeCircum);
            outputBox.printLine("Differenza: " +
                                (largeCircum – smallCircum));

      }
}
```

Esercizio 28

Scrivere un'applicazione per convertire centimetri in piedi e pollici (1piede = 12 pollici, 1 yard = 30 piedi). Si utilizzi un oggetto `InputBox` per la richiesta dati e un ogggetto `OutputBox` per la visualizzazione dei risultati. Si realizzi questa applicazione implementando una classe istanziabile i cui oggetti siano capaci di convertire da e per misure statunitensi.. Alcuni dei possibili metodi sono:

public double toFeet(double centimeter);
public double toInches(double centimeter);
public double toCentimeter(int feet, int inches);
public double toCentimeter(double yard);

Esercizio 29

Scrivere un'applicazione che calcola l'area di una corona circolare, dati i raggi della circonferenza interna e della circonferenza esterna. Si definisca a tal fine una classe istanziabile `Circle` che ha un metodo per calcolare l'area della circonferenza. Si preveda anche un metodo per impostare la lunghezza del raggio.

Esercizio 30

Quali dei seguenti costrutti `if` sono equivalenti?

```
a. if ( a ==  b )
        if (c == d)  a = 1;
        else b = 1;
b. if ( a == b) {
        if (c == d) a = 1;  }
        else b = 1;
```

c.
```
if  (a == b)
       if (c == d) a = 1;
  else b = 1;
```

Esercizio 31

Valutare le seguenti espressioni booleane. Per ciascuna delle seguenti espressioni si assuma che `x` sia `10`, `y` sia `20` e `z` sia `30`. Indicare quali delle seguenti espressioni sono sempre vere e quali sempre false, indipendentemente dai valori di `x`, `y`, `z`.

a. `x < 10 || x > 10`
b. `x  > y  && y > x`
c. `(x < y + z) && (x + 10 <= 20)`
d. `z – y == x && Math.abs(y – z) == x`
e. `x < 10 && x > 10`
f. `x > y || y > x`
g. `!(x < y + z) || !(x + 10 <= 20)`
h. `!(x == y) && (x != y) && (x < y || y < x)`

Esercizio 32

Esprimere il seguente costrutto `switch` utilizzando degli `if` annidati.

```
switch (grade) {
     case 10 :
     case  9 : a = 1;
            b = 2;
            break;
     case 8 : a = 3;
           b = 4;
           break;
     deafault : a = 5;
              break;
}
```

Esercizio 33

Scrivere un costrutto `if` per trovare il più piccolo di tre interi senza usare il metodo `min` della classe `Math`.

Esercizio 34

Rappresentare il flusso di esecuzione del programma per i due costrutti `switch` seguenti.

```
switch (choice) {
  case 1: a = 0;
          break;
  case 2: b = 1;
          break;
  case 3: c = 2;
          break;
  default : d = 3;
            break;
}
```

```
switch (choice) {
  case 1: a = 0;

  case 2: b = 1;

  case 3: c = 2;

  default : d = 3;
}
```

Esercizio 35

Scrivere un'applicazione che visualizzi l'espressione relativa ad un particolare anno usando i numeri romani. Per esempio, 1972 diventa MCMLXXII. Si utilizzi la seguente tabella:

Numero romano	Numerazione araba
I	1
V	5
X	10
L	50
C	100
D	500
M	1000

Si ricordi che certi numeri si esprimono per sottrazione, ad es. 4 si esprime con IV, 9 con IX, 400 con CD, ecc. 900 con CM, ecc.

Esercizio 36

In un ristorante l'ordinazione ai tavoli è automatizzata mediante un programma che consente all'utente di scegliere primo, secondo, dessert e bevanda. Le scelte con i relativi prezzi, sono le seguenti:

PRIMI		SECONDI		DESSERT		BEVANDE	
Ravioli al ragù	5 euro	Torta pasqualina	6 euro	Tiramisù	3 euro	Acqua	1 euro
Pansotti in salsa di noci	5.5 euro	Stoccafisso alla Genovese	5 euro	Torta di mele	2.5 euro	Vino rosso	2 euro
Trenette al pesto	4 euro	Polpo in aglio e prezzemolo	4.5 euro	Torta di pinoli	3 euro	Vino bianco	2 euro
Minestrone	3.5 euro	Cima	4.5 euro	Torta di rose	2.5 euro	Birra	1.5 euro

Si utilizzino degli oggetti `ListBox` per mostrare le voci dei menu divise per categoria. Si utilizzi `OutputBox` per mostrare il prezzo totale dell'ordine (considerando un costo aggiuntivo di 1.5 euro per pane e coperto).

Esercizio 37

Un servizio di nolleggio motorini applica le seguenti tariffe:

Tipo motorino	giorni feriali	weekend
vespino 50	15 euro per le prime tre ore, 2.5 euro all'ore per le ore successive	30 euro per le prime tre ore, 7.50 euro all'ora per le ore successive
vespone 200	25 euro per le prime tre ore, 3,5 euro all'ora per le ore successive	35 euro per le prime tre ore, 8.50 euro all'ora per le ore successive

Esercizio 38

Identificare tutti gli errori nei seguenti cicli. Alcuni errori sono sintattici, mentre altri sono logici (ad es. cicli infiniti).

a.
```
for (int i = 10; i > 0; i++) {
      x = y;
      a = b;
}
```
b.
```
sum = 0;
do {
    num = inputBox.getInteger( );
    sum += num;
} until (sum > 10000);
```
c.
```
while (x < 1 && x > 10) {
      a = b;
}
```
d.
```
while (a == b);
{
      a = b;
      x = y;
}
```
e.
```
for (int i = 1.0; i <= 2.0; i += 0.1) {
      x = y;
      a = b;
}
```

Esercizio 39

Scrivere cicli for, do-while, e while per calcolare le seguenti somme e prodotti.

a. $1 + 2 + 3 + ... N$
b. $5 + 10 + 15 + ... + 5N$
c. $1 + 3 + 7 + 15 + 31 + ... + (2^N - 1)$
d. $1 + 1/2 + 1/3 + 1/4 + ... + 1/N$
e. $1 * 2 * 3 * ... * N$
f. $1 * 2 * 4 * 8 * ... * 2^N$

Esercizio 40

Qual è il valore di sum al termine dell'esecuzione dei seguenti cicli?

a.
```
sum = 0;
for (int i = 0; i <= 10; i++)
      for (int j = 0; j <= 10; j++)
            sum += i;
```

```
b. sum = 0;
   j = 0;
   do {
        j++;
        for (int i = 5; i  > j; i--)
             sum = sum + (i + j);
   } while (j < 11);
c. sum = 0;
   i = 0;
   while (i < 5) {
        j = 5;
        while (i != j) {
             sum += j;
             j--;
        }
   }
d. sum = 0;
   for (int i = 0; i <= 10; i++)
        for (int j = 10; j > 2*i; j--)
             sum = sum + (j – i);
```

Esercizio 50

Scrivere un'applicazione per stampare i numeri da 1 a N nel seguente modo:

1 2 3 4 5 6 7 8 9 10
11 12 13 14 15 16 17 18 19 20
21 22 23 ...

L'applicazione può essere scritta in due modi diversi:

- utilizzando due cicli annidati
- utilizzando un singolo ciclo e l'operazione di modulo %

Esercizio 51

Un numero primo è un numero maggiore di uno e divisibile solo per se stesso e uno. Ad esempio, 2, 3, 5, 7 e 11 sono numeri primi. Scrivere un metodo che restituisce true se l'argomento è un numero primo. Scrivere un altro metodo che stampa tutti i fattori di un numero quando questo non è primo.

Esercizio 52

Un numero perfetto è un intero positivo che è uguale alla somma dei suoi divisori propri, cioè diversi dal numero stesso. Ad esempio, 6 è un numero perfetto perchè 1+2+3=6 e 1,2,3 sono tutti e i soli divisori propri di 6, mentre 8 non è un numero perfetto perché

1+2+4≠8. Scrivere un’applicazione che accetta un numero intero positivo e determina se è perfetto o meno.

Esercizio 53

Scrivere un’applicazione che visualizzi tutti i numeri perfetti tra 6 e N, N scelto dall’utente. Dato che ci sono pochi numeri perfetti, è meglio visualizzare anche i numeri che vengono scartati (ad esempio su una finestra distinta) in modo da essere certi che il programma sta continuando ad eseguire i suoi compiti.

Esercizio 54

Si modifichi il programma dell’esercizio 36 in modo da consentire al cliente di saltare il primo, il secondo o il dessert. Utilizzare un `ResponseBox` per chiedere all’utente se desidera o meno una particolare portata prima di proporgli una scelta.

Esercizio 55

Estendere il programma dell’esercizio 54 in modo che l’utente possa ordinare più di un piatto per ogni portata. Ad esempio, dovrebbe essere possibile ordinare due piatti di Trenette al Pesto, ecc.

Esercizio 56

Scrivere un metodo che ritorni il numero di lettere maiuscole in un oggetto String passato al metodo come parametro. Il metodo di classe `isUpperCase` della classe `Character` restituisce true se il carattere passato come argomento è maiuscolo. Utilizzando il manuale del linguaggio, si esplorino le possibilità della classe `Character` nel package `java.lang`.

Esercizio 57

Riscrivere il metodo dell’esercizio 56 senza utilizzare la classe Character.

Esercizio 58

Scrivere i seguenti metodi che manipolano stringhe passate come argomento:

- un metodo che restituisce una stringa con tutte le maiuscole cambiate in minuscole e tutte le minuscole cambiate in maiuscole;

- un metdo che restituisce la stringa in ordine inverso;
- un metodo che traspone le parole, ad esempio “Per tutti quanti” diventa “reP ittut itnauq”; per semplificare il problema si può assumere che la stringa in ingresso non contenga segni di punteggiatura; si può anche assumere che la stringa in ingresso cominci con un carattere e che vi sia esattamente uno spazio tra le parole;
- un metodo che ritorna true se la stringa è palindroma, ossia si legge uguale sia da sinistra a destra, sia da destra a sinistra. Ad esempio, osso e ala sono parole palindrome.

Esercizio 60

Scrivere una variazione del gioco Eggy-Peggy (vedi dispense del corso). Realizzare le seguenti varianti:

- Sha, aggiunge “sha” all’inizio di ogni parola
- Na, aggiunge “na” alla fine di ogni parola
- Sha Na Na, aggiunge “sha” all’inizio di ogni parola e “nana” alla fine di ogni parola.
- Ava, sposta la prima lettera al termine della parola e aggiunge “ava”.

Utilizzare un oggetto `ListBox` per far scegliere all’utente una delle varianti. Utilizzare un oggetto `InputBox` per leggere una frase (Suggerimento: ingrandire l’oggetto `InputBox` utilizzando il suo metodo resize).

Esercizio 61

Scrivere un’applicazione che chieda all’utente una frase e visualizzi il numero totale di ogni vocale minuscola presente nella frase. Utilizzare un oggett `OutputBox` per visualizzare il risultato.

Esercizio 62

Il gioco di parole Eggy-Peggy è un esempio di codifica cifrata. Un altro esempio (più serio) di codifica cifrata è la seguente: ogni carattere nel messaggio originale è spostato di N posizioni. Per esempio se N=1 il messaggio “Ciao a tutti” diventa “Djbp!b!uvuuj”.
Il messaggio cifrato viene decifrato riportando ogni carattere indietro di N posizioni. Lo spostamento di N posizioni è ottenuto convertendo il carattere in codice ASCII e aggiungendo o sottraendo N. Scrivere un’applicazione che legge il messaggio originale e visualizza il testo cifrato. Assicurarsi che il carattere ASCII che risulta dalla codifica ricada comunque tra 32 e 126. Ad esempio se N=8, e il carattere è la “z” (122), si deve ottenere come codifica 36 (122 + 8 = 130 – 126 = 4 da cui 32 + 4 = 36).

Esercizio 63

Scrivere un’applicazione che legge un testo cifrato con il metodo dell’esercizio precedente e decifra il testo mostrando il messaggio originale.

Esercizio 64

Un'altra tecnica di codifica è il codice Vignere. Questa tecnica è simile a quella dell'esercizio 62 dato che una chiave è applicata ciclicamente al messaggio originale. Per ora assumiamo che una chiave sia composta da sole lettere maiuscole. La codifica avviene aggiungendo ai codici dei caratteri del messaggio da cifrare, i valori dei codici corrispondenti ai caratteri della chiave. I codici ai caratteri della chiave sono assegnati come segue: 0 per A, 1 per B, e così via. Supponiamo che la chiave sia CIAO. La codifica funziona come segue:

```
Un messaggio da codificare
+
CIAOCIAOCIAOCIAOCIAOCIAOCI
=
Zw ....
```

Scrivere un classe istanziabile che:

- consenta di impostare un valore per la chiave
- metta a disposizione metodi per la codifica e la decodifica di messaggi segreti

Utilizzare la stessa tecnica dell'esercizio 62 per evitare di visualizzare caratteri al di fuori della gamma usuale.

Esercizio 65

Si identifichino i problemi nel seguente codice sorgente:

```
public int searchAccount( int[25] number )
{
     number = new int[15];
     for (int i = 0; i < number.length; i++ )
          number[i] = number[i –1] + number[i + 1];
     return number;
}
```

Esercizio 66

Si identifichino i problemi nel seguente codice sorgente:

```
class Q2
{
     private int alpha;
     private int beta;

     public static void classMethod( )
     {
          this.beta = this.alpha * 2;
     }
```

```
    public Q2( )
    {
        Q2( 0, 0 );
    }

    public Q2( int x, int y )
    {
        alpha = this.x;
        beta = this.y;
    }
}
```

Esercizio 67

Dichiarare un vettore di 365 float per contenere le temperature giornaliere di un anno. Utilizzando questa struttura dati all'interno di una classe istanziabile, scrivere i seguenti metodi:

- Impostazione e modifica della temperatura per un dato giorno
- Impostazione delle temperature a valori casuali (tra 10 e 30 gradi)
- Visualizzazione di tutte le temperature
- Visualizzazione delle temperatura di un giorno specifico dati il mese e il giorno
- Calcolo del giorno più caldo e di quello più freddo
- Calcolo della temperatura media mensile
- Calcolo della differenza tra il giorno più caldo e il giorno più freddo in ogni mese

Esercizio 68

Ripetere l'esercizio 67 utilizzando un vettore bi-dimensionale di 12 righe e 31 colonne.

Esercizio 69

Ripetere l'esercizio 67 utilizzando un vettore di oggetti Mese. Fare in modo che ogni oggetto possa essere configurato in modo da contenere il numero esatto di giorni.

Esercizio 70

Si supponga di voler mantenere aggiornate la più alta e la più bassa temperatura per ogni anno. Che tipo di struttura dati sarebbe necessaria? Si descrivano i pro e i contro delle possibili alternative.

Esercizio 71

L'officina di un concessionario di auto necessita di tenere traccia degli appuntamenti richiesti dai clienti per collaudi, tagliandi e altri tipi di manutenzioni e riparazioni. Quale struttura dati è più appropriata? Un vettore standard o un oggetto vector? Si supponga che il numero di appuntamenti possibili sia fissato per ogni giorno. Che tipo di struttura dati è

appropriata? Quali sono i criteri utilizzati per decidere? Implementare il sistema di gestione degli appuntamenti, creando due classi istanziabili:

- la classe Appuntamento, che contiene i dati del cliente, dell'autovettura e il tipo di intervento da effettuare (collaudo, tagliando, riparazione) e la durata prevista in ore.
- la classe GestioneOfficina, che consente di aggiungere appuntamenti, ricercare tutti gli appuntamenti per un dato giorno, cancellare e modificare appuntamenti.

Esercizio 72

Scrivere un'applicazione che consente la gestione completa di un'agenda. L'utente del programma ha quattro opzioni: aggiungere i dati di una persona, cancellarli, modificarli e ricercare i dati di una persona dato il cognome. Si preveda la possibilità che la ricerca ritorni più persone (casi di omonimia), che il programma rifiuti di aggiungere una persona i cui dati coincidono con una persona già presente nell'agenda. Si preveda la possibilità di eseguire ricerce avanzate (per esempio per città e per nome).

Esercizio 73

Realizzare una classe per effettuare conversioni di valuta. La classe deve essere in grado di gestire la conversione tra valute multiple. Ad esempio, sarà provvista di un metodo exchange che funziona come segue:

```
euro = converter.exchange( "dollar", "euro", 250.0 );
```

per convertire 250 dollari in una somma corrispondente in euro. Si supponga anche di avere un metodo per impostare i tassi di cambio. Ad esempio:

```
converter.setRate( "dollar", 0.98 );
```

significa che 1 dollaro equivale a 0.98 euro. Naturalmente sarà necessario un vettore per tenere traccia dei diversi tassi di cambio.

Esercizio 74

Scrivere una classe chiamata Arithmetic per giochi numerici. Un oggetto Arithmetic propone N problemi che comportano l'esecuzione di addizioni, sottrazioni, divisioni e moltiplicazioni, per esempio 4 + 5, 5*3, ecc. Ogni problema include un operatore, e gli operandi sono limitati a cifre singole. Il valore N è impostato dal programmatore. L'oggetto tiene traccia del numero di problemi chiesti e del numero di problemi risolti correttamente. Si preveda la possibilità di ottenere il numero di risposte corrette.

Esercizio 75

Scrivere un applicazione per la gestione di libretti di risparmio. Il programma gestisce diversi libretti di risparmio. Per ogni libretto, l'utente può depositare una somma, prelevare una somma, e chiedere il saldo. Sia i prelievi, sia i depositi sono numerati in ordine crescente.