Aturan Praktikum:

- Keterlambatan maksimal 30 menit
- Hasil praktikum, selesai atau tidak selesai ditanggung mahasiswa
- Nilai praktikum adalah nilai perorangan, sebagai tambahan nilai test atau quiz jika nilai test atau quiz kurang mencukupi
- Mahasiswa login ke komputer
- Persiapan dan penjelasan praktikum 10 menit
- Praktikum dilakukan selama 70 menit terdiri dari praktikum sesuai modul dan sebuah soal praktikum
- Hasil praktikum di-zip dan dikumpulkan
- Sisa waktu 20 menit digunakan untuk tes kecil hasil praktikum, saat tes tampilan komputer adalah background desktop dan mahasiswa tidak diperkenankan menggunakan komputer dan melihat modul praktikum
- Mahasiswa logout dari komputer
- Praktikum selesai

JCE (Java Cryptoghraphy Extension)
(Minggu 10 – Praktikum 1)

1. Praktikum

a. Persiapan

- Membuat direktori kerja dengan nama SI319-P10-1-Kelas-NIM misalnya SI319-P10-1-A-23507024
- Di dalam direktori di atas, buat direktori JCE untuk menyimpan file-file yang akan dibuat.

b. Program enkripsi dengan JCE

i. DES.java

```
import java.security.*;
import javax.crypto.*;
import javax.crypto.spec.*;
import java.io.*;


public class DES {

      public static String asHex (byte buf[]) {
      // mengubah string sebagai bilangan hexadesimalnya

            StringBuffer strbuf = new StringBuffer(buf.length * 2);
            int i;
            for (i = 0; i < buf.length; i++) {
                  if (((int) buf[i] & 0xff) < 0x10)
```

```
                        strbuf.append("0");
                  strbuf.append(Long.toString(
                   (int) buf[i] & 0xff, 16));
            }

            return strbuf.toString();
      }

      public static void main(String[] args) throws Exception {
            System.out.println("DES - Data Encryption Standard\n");
            String message="Nama/NIM : [isi dengan nama/NIM]";
            // membuat key generator
            System.out.println("string sebelum dienkrip: " + message
                + "\n");
            KeyGenerator kgen = KeyGenerator.getInstance("DES");
            // key sebesar 56 bit
            kgen.init(56);

            // membuat spesifikasi kunci rahasia
            SecretKey skey = kgen.generateKey();
            byte[] raw = skey.getEncoded();
            SecretKeySpec skeySpec = new SecretKeySpec(raw, "DES");
            // menggenerasi chiper
            Cipher cipher = Cipher.getInstance("DES");

            // melakukan enkripsi
            cipher.init(Cipher.ENCRYPT_MODE, skeySpec);
            byte[] encrypted = cipher.doFinal(
              (args.length == 0 ? message : args[0]).getBytes());
            System.out.println("string yang dienkrip: "
              + asHex(encrypted) + "\n");

            // melakukan deskripsi
            cipher.init(Cipher.DECRYPT_MODE, skeySpec);
            byte[] original = cipher.doFinal(encrypted);
            String originalString = new String(original);
            System.out.println("string didekrip: " + originalString
              + " " + asHex(original) + "\n");
      }
}
```

ii. AES,java

```
import java.security.*;
import javax.crypto.*;
import javax.crypto.spec.*;
import java.io.*;


public class AES {

      public static String asHex (byte buf[]) {
      // mengubah string sebagai bilangan hexadesimalnya

            StringBuffer strbuf = new StringBuffer(buf.length * 2);
```

```
            int i;
            for (i = 0; i < buf.length; i++) {
                  if (((int) buf[i] & 0xff) < 0x10)
                        strbuf.append("0");
                  strbuf.append(Long.toString(
                    (int) buf[i] & 0xff, 16));
            }

            return strbuf.toString();
      }

      public static void main(String[] args) throws Exception {
            System.out.println("AES - Advanced Encryption Standard\n");
            String message="Nama/NIM : [isi dengan nama/NIM]";
            // membuat key generator
            System.out.println("string sebelum dienkrip: "
                + message + "\n");
            KeyGenerator kgen = KeyGenerator.getInstance("AES");
            // key sebesar 128 bit
            kgen.init(128);

            // membuat spesifikasi kunci rahasia
            SecretKey skey = kgen.generateKey();
            byte[] raw = skey.getEncoded();
            SecretKeySpec skeySpec = new SecretKeySpec(raw, "AES");
            // menggenerasi chiper
            Cipher cipher = Cipher.getInstance("AES");

            // melakukan enkripsi
            cipher.init(Cipher.ENCRYPT_MODE, skeySpec);
            byte[] encrypted = cipher.doFinal(
               (args.length == 0 ? message : args[0]).getBytes());
            System.out.println("string yang dienkrip: "
               + asHex(encrypted) + "\n");

            // melakukan deskripsi
            cipher.init(Cipher.DECRYPT_MODE, skeySpec);
            byte[] original = cipher.doFinal(encrypted);
            String originalString = new String(original);
            System.out.println("string didekrip: " + originalString
               + " " + asHex(original) + "\n");
      }
}
```

iii. Blowfish.java

```
import java.security.*;
import javax.crypto.*;
import javax.crypto.spec.*;
import java.io.*;


public class Blowfish {

      public static String asHex (byte buf[]) {
```

```
      // mengubah string sebagai bilangan hexadesimalnya

            StringBuffer strbuf = new StringBuffer(buf.length * 2);
            int i;
            for (i = 0; i < buf.length; i++) {
                  if (((int) buf[i] & 0xff) < 0x10)
                        strbuf.append("0");
                  strbuf.append(Long.toString(
                    (int) buf[i] & 0xff, 16));
            }

            return strbuf.toString();
      }

      public static void main(String[] args) throws Exception {
            System.out.println("Blowfish\n");
            String message="Nama/NIM : [isi dengan nama/NIM]";
            // membuat key generator
            System.out.println("string sebelum dienkrip: "
                + message + "\n");
            KeyGenerator kgen = KeyGenerator.getInstance("Blowfish");
            // key sebesar 128 bit
            kgen.init(128);

            // membuat spesifikasi kunci rahasia
            SecretKey skey = kgen.generateKey();
            byte[] raw = skey.getEncoded();
            SecretKeySpec skeySpec =
               new SecretKeySpec(raw, "Blowfish");
            // menggenerasi chiper
            Cipher cipher = Cipher.getInstance("Blowfish");

            // melakukan enkripsi
            cipher.init(Cipher.ENCRYPT_MODE, skeySpec);
            byte[] encrypted = cipher.doFinal(
              (args.length == 0 ? message : args[0]).getBytes());
            System.out.println("string yang dienkrip: "
              + asHex(encrypted) + "\n");

            // melakukan deskripsi
            cipher.init(Cipher.DECRYPT_MODE, skeySpec);
            byte[] original = cipher.doFinal(encrypted);
            String originalString = new String(original);
            System.out.println("string didekrip: " + originalString
              + " " + asHex(original) + "\n");
      }
}
```

iv. Menjalankan aplikasi

1. Lakukan kompilasi untuk semua file .java dengan perintah:

   `javac [nama_file.java]`

   misal:

```
javac DES.java
```

(
jika javac tidak dikenali maka set path dengan perintah:

```
path=c:/jdk/bin;%path%
```

)

2. lalu eksekusi semua program dengan perintah:

```
java –cp . [nama_kelas]
```

misal:

```
java –cp . DES
```

v. Laporan

- Lakukan eksplorasi mengenai JCE dan buat rangkuman JCE maksimal 1 halaman.
- Kesimpulan program
- Print screen hasil eksekusi program