Proyecto programado lll – Eiffel.

José Andrés Navarro Acuña.

2016254241.

Instituto Tecnológico de Costa Rica.

Ingenieria en computación.

IC-4700 Lenguajes de programación.

José Enrique Araya Monge.

Grupo 1.

I semestre 2018.

1. Introducción

El presente proyecto programado consistió en la elaboración en el lenguaje de programación Eiffel (Orientado a Objetos) el problema del empaque, el cual consiste en colocar objetos de diferentes tamaños dentro de cajas con una capacidad dada de almacenamiento, buscando minimizar el número de cajas requeridas. Sin embargo, resolver en forma óptima este problema es computacionalmente muy caro. Pero existen muchos algoritmos que aunque no son óptimos garantizan resultados muy buenos en la práctica. Se presentan por tanto, tres algoritmos de los cuales debieron ser implementados para simular el desempeño de dicho problema:

- **FFD (First Fit Decreasing):** Ordena los objetos por tamaño decreciente y busca colocar cada objeto en la primera caja disponible que tenga campo suficiente.
- **FF (First Fit):** No ordena los objetos por tamaño; simplemente los acomoda en el orden en que se presentan; busca la primera caja en donde pueda almacenarlos.
- **BF (Best Fit):** Tampoco ordena los objetos por tamaño; para cada objeto escoge la caja que esté más llena y en la que aún queda campo para el objeto.

2. Estructuras de clases usadas

En la elaboración del proyecto se implementaron las siguientes clases para dar con la solución al problema:

- **APPLICATION:** Corresponde con la clase que crea las instancias y ejecuciones de los tres algoritmos implementados (**FIRST_FIT, FIRST_FIT_DECREASING** y **BEST_FIT**). Además, es la encargada de solicitar y almacenar los 4 parametros que son requeridos por el usuario, es decir, el tamaño de las cajas, el tamaño máximo de los objetos, la semilla y el número de objetos a generar. Se encarga tambien, de generar la lista de objetos que serán probados en los algoritmos de acuerdo a los parámetros que inserta el usuario, es decir, crea objetos de tipo **OBJETO**.

- **CAJA:** Clase que contiene la lista de objetos de las posibles soluciones que se van generando acorde a los algoritmos. Esta clase está relacionada con la clase **FIT**, ya que, es la clase padre de los algoritmos, por tanto, se encarga de crear instancias de esta clase, **CAJA**. Los atributos de dicha clase corresponden con un identificador, el tamaño, el espacio disponible, y como se mencionó, la lista de objetos que contendrá.

- **OBJETO:** Clase que representa los objetos que son creados en la clase **APPLICATION** y que posteriormente, son manipulados por los distintos algoritmos para dar con las respectivas soluciones. Dicha clase contiene como atributos un identificador y un tamaño.

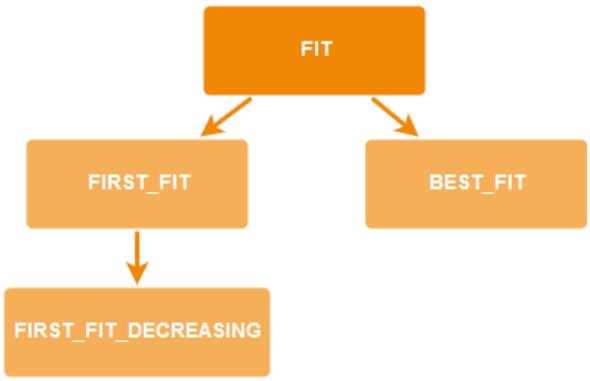
- **FIT:** Corresponde a la clase padre de los tres algoritmos, por tanto, hereda sus propiedades a las clases **FIRST_FIT, FIRST_FIT_DECREASING** y **BEST_FIT**. Tiene los siguientes atributos: tamaño de las cajas, la secuencia de objetos que generó la clase **APPLICATION** y la lista de la solución final del problema. Dicha clase crea objetos de tipo **CAJA**, para ser generados al momento que los algoritmos lo requieran.

- **FIRST_FIT:** Clase hija de la clase **FIT**. Esta clase unicamente implementa el algoritmo requerido como tal. Además, hereda a la clase **FIRST_FIT_DECREASING**, por tanto, se le provee a ésta clase el algoritmo que conlleva **FIRST_FIT**. Hace uso tambien, de las diversas rutinas que su clase padre le brinda.

- **FIRST_FIT_DECREASING:** Clase hija de la clase **FIRST_FIT**. Dicha clase unicamente implementa la rutina de ordenamiento de la secuencia de objetos debidamente por tamaño, posteriormente, ejecuta el algoritmo que le provee su clase padre.

- **BEST_FIT:** Clase hija de la clase **FIT**. Dicha clase implementa el algoritmo que éste requiere como tal, además, realiza uso de los atributos y rutinas que su clase padre le provee.

A continuación, se muestra una grafica para mostrar las herencias implementadas en los algoritmos.



3. Instrucciones para ejecutar el programa

Primeramente, es necesario tener instalado en el ordenador el programa Eiffel Studio en la versión 18.01, posteriormente, basta con ejecutar dicho ambiente y seleccionar la pestaña **File** y el submenú **Open Proyect…**, seguidamente se debe buscar la ubicación del proyecto y seleccionar el archivo con extención **.ecf**, una vez cargado, se debe presionar la tecla **F5** para ejecutar el programa. Una vez realizado este paso, se le preguntará al usuario los parametros que desea ingresar, es decir, el tamaño de las cajas, el tamaño máximo de los objetos, la semilla y la cantidad de objetos que se deseen generar. Existe la posiblidad de establecer el valor default con tan solo insertar **-1** (menos uno). Además, cualquier valor que se considere como inválido será rechazado (caracteres y demás símbolos) y se le pedirá nuevamente que ingrese el valor. Una vez insertado los cuatro valores, inmediatamente se ejecutarán los algoritmos mostrando todos los datos que estos involucraron, tales como el listado de los objetos que se generaron, la visualización de las cajas, la cantidad de cajas requeridas, el promedio de ocupación de las cajas, entre otros puntos.

Un ejemplo de ejecución:

```
- Problema del empaque -

Ingrese el tamano de las cajas: (-1 valor default: 10)

-1
- Valor default establecido!
Ingrese el tamano maximo de los objetos: (-1 valor default: 7)

8
- Se ha establecido el valor!
Ingrese el valor de la semilla: (-1 valor default: 2018)

1997
- Se ha establecido el valor!
Ingrese el numero de objetos: (-1 valor default: 20)

wasd
- Error!, indique un valor valido
Ingrese el numero de objetos: (-1 valor default: 20)

-1
- Valor default establecido
```

Posteriormente, se mostrarán todos los datos correspondientes a los algoritmos.

4. Corridas de ejemplo

Corrida 1: (valores default)

- tamaño_cajas: 10, tamaño_max_objetos: 7, semilla: 2018, número_objetos: 20

```
Datos generales:
- Tamanio de las cajas: 10
- Tamanio maximo de los objetos: 7
- Semilla: 2018
- Numero de objetos generados: 20
- Suma del tamanio de los objetos: 93

Listado de objetos: (id, tamanio)
 (1, 1)  (2, 2)  (3, 7)  (4, 5)  (5, 2)  (6, 6)  (7, 2)  (8, 6)  (9, 5)  (10, 7)
 (11, 3)  (12, 7)  (13, 5)  (14, 7)  (15, 6)  (16, 7)  (17, 2)  (18, 1)  (19, 6)  (20, 6)
```

```
*** First Fit Decreasing (FFD): ***
1001: [ (3, 7)  (11, 3) ]
1002: [ (10, 7)  (7, 2)  (1, 1) ]
1003: [ (12, 7)  (5, 2)  (18, 1) ]
1004: [ (14, 7)  (17, 2) ]
1005: [ (16, 7)  (2, 2) ]
1006: [ (6, 6) ]
1007: [ (8, 6) ]
1008: [ (15, 6) ]
1009: [ (19, 6) ]
1010: [ (20, 6) ]
1011: [ (13, 5)  (4, 5) ]
1012: [ (9, 5) ]

 Cajas requeridas: 12
 Promedio de ocupacion: 0.77500000000000002
```

```
*** First Fit (FF): ***
1001: [ (1, 1)  (2, 2)  (3, 7) ]
1002: [ (4, 5)  (5, 2)  (7, 2)  (18, 1) ]
1003: [ (6, 6)  (11, 3) ]
1004: [ (8, 6)  (17, 2) ]
1005: [ (9, 5)  (13, 5) ]
1006: [ (10, 7) ]
1007: [ (12, 7) ]
1008: [ (14, 7) ]
1009: [ (15, 6) ]
1010: [ (16, 7) ]
1011: [ (19, 6) ]
1012: [ (20, 6) ]

 Cajas requeridas: 12
 Promedio de ocupacion: 0.77500000000000002
```

```
*** Best Fit (BF) ***
1001: [ (1, 1)  (2, 2)  (3, 7) ]
1002: [ (4, 5)  (5, 2)  (7, 2)  (18, 1) ]
1003: [ (6, 6) ]
1004: [ (8, 6) ]
1005: [ (9, 5)  (13, 5) ]
1006: [ (10, 7)  (11, 3) ]
1007: [ (12, 7)  (17, 2) ]
1008: [ (14, 7) ]
1009: [ (15, 6) ]
1010: [ (16, 7) ]
1011: [ (19, 6) ]
1012: [ (20, 6) ]

 Cajas requeridas: 12
 Promedio de ocupacion: 0.77500000000000002
```

Corrida 2:

- tamaño_cajas: 12, tamaño_max_objetos: 7, semilla: 2018, número_objetos: 20

```
Datos generales:
- Tamanio de las cajas: 12
- Tamanio maximo de los objetos: 7
- Semilla: 2018
- Numero de objetos generados: 20
- Suma del tamanio de los objetos: 93

Listado de objetos: (id, tamanio)
 (1, 1)  (2, 2)  (3, 7)  (4, 5)  (5, 2)  (6, 6)  (7, 2)  (8, 6)  (9, 5)  (10, 7)
 (11, 3)  (12, 7)  (13, 5)  (14, 7)  (15, 6)  (16, 7)  (17, 2)  (18, 1)  (19, 6)  (20, 6)
```

```
*** First Fit Decreasing (FFD): ***
1001: [ (3, 7)  (13, 5) ]
1002: [ (10, 7)  (4, 5) ]
1003: [ (12, 7)  (9, 5) ]
1004: [ (14, 7)  (11, 3)  (7, 2) ]
1005: [ (16, 7)  (5, 2)  (17, 2)  (1, 1) ]
1006: [ (6, 6)  (8, 6) ]
1007: [ (15, 6)  (19, 6) ]
1008: [ (20, 6)  (2, 2)  (18, 1) ]

 Cajas requeridas: 8
 Promedio de ocupacion: 0.96875
```

```
*** First Fit (FF): ***
1001: [ (1, 1)  (2, 2)  (3, 7)  (5, 2) ]
1002: [ (4, 5)  (6, 6)  (18, 1) ]
1003: [ (7, 2)  (8, 6)  (11, 3) ]
1004: [ (9, 5)  (10, 7) ]
1005: [ (12, 7)  (13, 5) ]
1006: [ (14, 7)  (17, 2) ]
1007: [ (15, 6)  (19, 6) ]
1008: [ (16, 7) ]
1009: [ (20, 6) ]

 Cajas requeridas: 9
 Promedio de ocupacion: 0.86111111111111116
```

```
*** Best Fit (BF) ***
1001: [ (1, 1)  (2, 2)  (3, 7)  (5, 2) ]
1002: [ (4, 5)  (6, 6)  (18, 1) ]
1003: [ (7, 2)  (8, 6)  (11, 3) ]
1004: [ (9, 5)  (10, 7) ]
1005: [ (12, 7)  (13, 5) ]
1006: [ (14, 7)  (17, 2) ]
1007: [ (15, 6)  (19, 6) ]
1008: [ (16, 7) ]
1009: [ (20, 6) ]

 Cajas requeridas: 9
 Promedio de ocupacion: 0.86111111111111116
```

Corrida 3:

- tamaño_cajas: 12, tamaño_max_objetos: 8, semilla: 2019, número_objetos: 25

```
Datos generales:
- Tamanio de las cajas: 12
- Tamanio maximo de los objetos: 8
- Semilla: 2019
- Numero de objetos generados: 25
- Suma del tamanio de los objetos: 119

Listado de objetos: (id, tamanio)
 (1, 6)  (2, 5)  (3, 1)  (4, 5)  (5, 2)  (6, 3)  (7, 7)  (8, 5)  (9, 5)  (10, 7)
 (11, 7)  (12, 8)  (13, 2)  (14, 4)  (15, 5)  (16, 3)  (17, 1)  (18, 3)  (19, 7)  (20, 3)
 (21, 8)  (22, 6)  (23, 5)  (24, 6)  (25, 5)
```

```
*** First Fit Decreasing (FFD): ***
1001: [ (12, 8)  (14, 4) ]
1002: [ (21, 8)  (18, 3)  (17, 1) ]
1003: [ (7, 7)  (4, 5) ]
1004: [ (10, 7)  (15, 5) ]
1005: [ (11, 7)  (2, 5) ]
1006: [ (19, 7)  (8, 5) ]
1007: [ (1, 6)  (22, 6) ]
1008: [ (24, 6)  (23, 5)  (3, 1) ]
1009: [ (9, 5)  (25, 5)  (13, 2) ]
1010: [ (6, 3)  (20, 3)  (16, 3)  (5, 2) ]

 Cajas requeridas: 10
 Promedio de ocupacion: 0.9916666666666667
```

```
*** First Fit (FF): ***
1001: [ (1, 6)  (2, 5)  (3, 1) ]
1002: [ (4, 5)  (5, 2)  (6, 3)  (13, 2) ]
1003: [ (7, 7)  (8, 5) ]
1004: [ (9, 5)  (10, 7) ]
1005: [ (11, 7)  (14, 4)  (17, 1) ]
1006: [ (12, 8)  (16, 3) ]
1007: [ (15, 5)  (18, 3)  (20, 3) ]
1008: [ (19, 7)  (23, 5) ]
1009: [ (21, 8) ]
1010: [ (22, 6)  (24, 6) ]
1011: [ (25, 5) ]

 Cajas requeridas: 11
 Promedio de ocupacion: 0.90151515151515149
```

```
*** Best Fit (BF) ***
1001: [ (1, 6)  (2, 5)  (3, 1) ]
1002: [ (4, 5)  (5, 2)  (6, 3)  (13, 2) ]
1003: [ (7, 7)  (8, 5) ]
1004: [ (9, 5)  (10, 7) ]
1005: [ (11, 7)  (15, 5) ]
1006: [ (12, 8)  (14, 4) ]
1007: [ (16, 3)  (17, 1)  (18, 3)  (20, 3) ]
1008: [ (19, 7)  (23, 5) ]
1009: [ (21, 8) ]
1010: [ (22, 6)  (24, 6) ]
1011: [ (25, 5) ]

 Cajas requeridas: 11
 Promedio de ocupacion: 0.90151515151515149
```

5. Comentarios finales

El estado final del proyecto es del 100%, es decír, todas las funcionalidades solicitadas fueron implementadas, siguiendo cuidadosamente los requisitos propuestos en el enunciado del proyecto; la abstracción, el encapsulamiento, la modularidad y la jerarquia de herencia, es decir, los principios generales de la orientación a objetos, además, se implementó el principio de diseño por contrato en aquellas rutinas donde se consideró necesario. Con respecto a los problemas encontrados y las limitaciones, practicamente ninguno.