

Semana 3

Fundamentos de Programación (PRY2201)

Pauta de evaluación sumativa

# Problemas Algorítmicos con Expresiones aritméticas y lógicas

A continuación, se presentan los descriptores de los niveles de logro con los que serás retroalimentado/a y/o evaluado/a:

| CL | Completamente Logrado (100%) | Logro **completo** del manejo del indicador, se considera como el punto **óptimo**, de acuerdo al área evaluada, dentro del rango de competente |
| --- | --- | --- |
| L | Logrado (80%) | Logro del indicador, se observan **pequeñas dificultades o errores** para alcanzar el logro total y sobresaliendo de lo mínimo aceptable |
| ML | Medianamente Logrado (60%) | Logro **mínimo aceptable** en el manejo del indicador que permite ser considerado competente |
| LI | Logro Insuficiente (30%) | **Logro insuficiente** en el manejo del indicador, se observan **varias dificultades o errores** que no le permiten alcanzar el logro mínimo para ser considerado competente |
| NL | No Logrado (0%) | Se observan un **escaso, nulo o incorrecto** logro en el manejo del indicador |

| N° | Criterios | Completamente Logrado (100%) | **Logrado (80%)** | Medianamente Logrado (60%) | **Logro Insuficiente** **(30%)** | **No Logrado** **(0%)** |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | Construye un algoritmo identificando las entradas, procesos y salidas para dar solución al problema planteado.  (10 Puntos) | Construye un algoritmo identificando las entradas, procesos y salidas, proporcionando una solución integral al problema planteado. | Construye un algoritmo que identifica las entradas, procesos y salidas, con pequeñas deficiencias en la claridad o integralidad de la solución planteada. | Construye un algoritmo que identifica algunas entradas, procesos y salidas, pero su claridad o integralidad son limitadas o ineficientes, dificultando el proponer una solución al problema planteado. | Construye un algoritmo, pero solo identifica la entrada, afectando la claridad e integralidad de la solución al problema planteado. | No identifica las entradas, procesos y salidas del problema planteado. |
| 2 | Selecciona una estrategia de abstracción acorde a la necesidad de la problemática planteada.  (10 puntos) | Selecciona una estrategia de abstracción cuya elección brinda una solución acorde a la problemática planteada. | Selecciona una estrategia de abstracción, aunque la justificación es menos clara o precisa, dando solución a la problemática con algunos reparos. | Selecciona una estrategia de abstracción, pero la elección y la justificación son limitadas o poco claras para dar solución a la problemática planteada. | Selecciona una estrategia de abstracción  poco adecuada para dar solución a la problemática planteada. | No selecciona ninguna estrategia de abstracción para el desarrollo de su algoritmo. |
| 3 | Utiliza variables en el desarrollo de la información en su algoritmo y en el código Java.  (15 puntos) | Utiliza 4 o más variables, asegurando una gestión eficiente de la información en el pseudocódigo y en el código Java. | Utiliza 3 variables, aunque con deficiencias en la gestión de la información en el pseudocódigo y en el código Java. | Utiliza 2 variables, y su gestión es limitada en el pseudocódigo y en el código Java. | Utiliza solo una variable en el desarrollo de su algoritmo, ocasionando una gestión deficiente en el pseudocódigo y en el código Java. | No utiliza variables en el pseudocódigo ni en el código Java. |
| 4 | Aplica expresiones aritméticas para desarrollar un algoritmo y en el código Java.  (15 puntos) | Aplica 3 o más expresiones aritméticas de manera correcta en el pseudocódigo y en el código Java. | Aplica al menos 3 expresiones aritméticas, aunque presenta deficiencias en la lógica o la implementación en el pseudocódigo y en el código Java. | Aplica 2 expresiones aritméticas, pero su lógica o implementación son limitadas o ineficientes en el pseudocódigo y en el código Java. | Aplica 1 expresión aritmética deficiente, afectando significativamente la lógica en el pseudocódigo y en el código Java. | No aplica adecuadamente expresiones aritméticas en el pseudocódigo ni en el código Java. |
| 5 | Aplica las expresiones lógicas para desarrollar un algoritmo en pseudocódigo y en el código Java.  (15 puntos) | Aplica todas las expresiones lógicas de manera coherente en el desarrollo del algoritmo en pseudocódigo y en el código Java. | Aplica todas las expresiones lógicas, pero presenta algunas deficiencias en la coherencia o la implementación en el desarrollo del algoritmo en pseudocódigo y en el código Java. | Aplica algunas expresiones lógicas, pero su coherencia o implementación son limitadas o ineficientes en el desarrollo del algoritmo en pseudocódigo y en el código Java. | Aplica una expresión lógica, pero su aplicación es deficiente, afectando significativamente la coherencia en el desarrollo del algoritmo en pseudocódigo y en el código Java | No aplica expresiones lógicas en el desarrollo de su algoritmo y en el código Java. |
| 6 | El algoritmo permite que los datos se soliciten e ingresen por pantalla y muestra el valor de la tarifa a pagar.  (15 puntos) | Permite que los datos se soliciten e ingresen por pantalla y muestra de manera clara el valor de la tarifa a pagar en el pseudocódigo y en el código Java. | Permite que los datos se ingresen por pantalla y muestra el valor de la tarifa a pagar, aunque puede haber pequeñas deficiencias en la implementación en el pseudocódigo y en el código Java. | Permite que los datos se ingresen por pantalla, pero la visualización de la tarifa es limitada o ineficiente en el pseudocódigo y en el código Java. | La solicitud e ingreso de datos por pantalla y la visualización de la tarifa son deficientes, afectando significativamente la claridad en el pseudocódigo y en el código Java. | No permite adecuadamente la solicitud e ingreso de datos por pantalla ni muestra de manera clara el valor de la tarifa en el pseudocódigo ni en el código Java. |
| 7 | El algoritmo aplica descuentos, dependiendo del tipo de entrada.  (10 puntos) | Aplica descuentos de manera correcta dependiendo del tipo de entrada en el pseudocódigo y en el código Java. | Aplica descuentos dependiendo del tipo de entrada, aunque puede haber pequeñas deficiencias en la implementación en el pseudocódigo y en el código Java. | Aplica descuentos dependiendo del tipo de entrada, pero la implementación es limitada o ineficiente en el pseudocódigo y en el código Java. | La aplicación de descuentos dependiendo del tipo de entrada es deficiente, afectando significativamente la precisión en el pseudocódigo y en el código Java. | No aplica adecuadamente descuentos dependiendo del tipo de entrada en el pseudocódigo ni en el código Java. |
| 8 | El algoritmo calcula el total a pagar, de acuerdo a la compra realizada.  (10 puntos) | Calcula correctamente el total a pagar de acuerdo a la compra realizada en el pseudocódigo y en el código Java. | Calcula el total a pagar de acuerdo a la compra realizada, aunque puede haber pequeñas deficiencias en la implementación en el pseudocódigo y en el código Java. | Calcula el total a pagar de acuerdo a la compra realizada, pero la implementación es limitada o ineficiente en el pseudocódigo y en el código Java. | El cálculo del total a pagar de acuerdo a la compra realizada es deficiente, afectando significativamente la precisión en el pseudocódigo y en el código Java. | No calcula adecuadamente el total a pagar de acuerdo a la compra realizada en el pseudocódigo ni en el código Java. |



Reservados todos los derechos Fundación Instituto Profesional Duoc UC. No se permite copiar, reproducir, reeditar, descargar, publicar, emitir, difundir, de forma total o parcial la presente obra, ni su incorporación a un sistema informático, ni su transmisión en cualquier forma o por cualquier medio (electrónico, mecánico, fotocopia, grabación u otros) sin autorización previa y por escrito de Fundación Instituto Profesional Duoc UC La infracción de dichos derechos puede constituir un delito contra la propiedad intelectual.