

Programación Móvil

Dr. Marco Aurelio Nuño Maganda

Universidad Politecnica de Victoria
Ingeniería en Tecnologías de la Información
Cuatrimestre Septiembre - Diciembre 2024

mnunom@upv.edu.mx

August 29, 2024

Breve CV del Facilitador

- Doctor en Ciencias Computacionales por parte del INAOE (2009).
- Profesor de Tiempo Completo de la UPV desde 2009.
- Miembro del Sistema Nacional de Investigadores - Nivel Candidado (2014-2016), Nivel I (2020-2022), Nivel I (2023-2027)
- 17 tesis dirigidas a nivel maestría.
- Asignaturas impartidas en el pasado
 - Licenciatura: Cómputo en Dispositivos Móviles, Graficación por Computadora Avanzada, Lenguajes y Automatas, Programación Orientada a Objetos
 - Maestría: Visión por computadora, Tópicos Selectos de Imagenología, Fundamentos de Sistemas de Información
- Miembro del Núcleo Académico Básico (NAB) de la maestría en Ingeniería de la UPV.

Plataforma Virtual para el Curso

- Nombre de la clase: **Programación Móvil-Septiembre - Diciembre 2024**
- Código de clase en Classroom: **xv543pb**
- Enlace Meet para sesiones no presenciales:
<https://meet.google.com/jsb-nmxy-dht>

Horario de la Clase

■ Días y horas de clase

Clave de Grupo	Lunes	Martes	Miércoles	Jueves	Viernes
ITI-271311	14:00- 14:54	8:50- 10:40	14:00-14:54	9:45 - 10:40	12:05 - 13:00
ITI-271304		11:10 -13:00	9:45-12:05	11:10 -13:00	

■ Fechas Importantes:

- Inicio de Cursos: 2/Septiembre
- Fin de Cursos: 13/Diciembre
- Dias no hábiles oficiales: 16 de septiembre (lunes), 1 de octubre (martes) y 18 de noviembre (lunes).

Plataforma Virtual para el Curso

- Nombre de la clase: **Programación Móvil- Septiembre - Diciembre 2024**
- Código de clase en Classroom: **xv543pb**
- Enlace Meet para sesiones no presenciales:
<https://meet.google.com/jsb-nmxy-dht>

Reglas básicas

- Se recomienda puntualidad y asistencia a las sesiones.
- Respeto hacia el profesor y hacia sus compañeros y compañeras.
- No se permite el ingreso y/o ingestión de **Alimentos** ni **Bebidas** de ningún tipo a la clase.
- Solo se puede usar **AUDIFONOS O DISPOSITIVOS MANO-LIBRES EN CLASE** previa autorización por parte del profesor. Cualquier uso no AUTORIZADO es motivo de amonestación al estudiante, y expulsión en caso de reincidir sin derecho a réplica.

Uso del Teléfono Inteligente

- Se recomienda no utilizarlo durante el transcurso de la clase. Depende del comportamiento del grupo que esto no sea aplicado...

Resguardo del teléfono inteligente

De ser necesario, se solicitará al INICIO de la CLASE a todos los asistentes a la clase (incluyendo al profesor) guardar su telefono en una caja, la cual será cerrada, regresando su telefono al finalizar la SESION.



Pase de Lista

- Se pasa lista al inicio de la clase. En caso de reincorporación tardía, se pone un retardo.
- DOS RETARDOS equivalen a una INASISTENCIA, que no es JUSTIFICABLE.
- Al NO alcanzar un 80% de asistencia, el estudiante pierde su derecho de ser EVALUADO.
- Se deja un margen del 20% de inasistencia que el estudiante puede manejar a lo que mas le convenga.
- 15 semanas, 3 sesiones por semana, 45 sesiones (8 inasistencias al cuatrimestre por motivos meramente lúdicos, personales o por tragedias [el vocho te va a dejar tirado por lo menos 4 veces al año.]).

Justificacion de Inasistencia (1)

Para justificar una inasistencia, es necesario cumplir con los siguientes pasos:

- Ingresar al apartado de Google classroom creado para dicho fin y cargar un archivo PDF para cada día (o lapso) de inasistencias a justificar.
- Formato del nombre del PDF (todo en minúsculas):

iti-<clavegrupo>-<apat>-<amat>-<nombre>-<dd1>-<mm1>-<yyyy1>-<dd2>-<mm2>-<yyyy2>.pdf

- Donde (Si pone otros datos, dicha justificacion queda anulada):
 - <clavegrupo> es la clave que viene en su horario.
 - <apat>-<amat>-<nombre> son sus nombres. Debe utilizar guiones bajo para sustituir los espacios en su datos (ver ejemplo).
 - <dd1>-<mm1>-<yyyy1> es la fecha de inicio de inasistencia
 - <dd2>-<mm2>-<yyyy2> es la fecha de fin de inasistencia (si es faltó solo día, use la misma 02-09-2024-02-09-2024)

■ Ejemplos

iti-552244-nuño-maganda-marco_aurelio-02-09-2024-02-09-2024.pdf

iti-001233-del_sagrado_corazon-hernandez-michael_jackson-02-09-2024-02-10-2024.pdf

Justificación de Inasistencia (2)

- El interior de ese PDF debe una digitalización del documento que justifique su inasistencia.
- Solo se reciben inasistencias por motivos médicos (receta legible con su Nombre) y de trabajo (Citas al SAT, Pasaporte, VISA, entrevista de trabajo).
- Debe venir resaltado el nombre y el motivo de inasistencia.
- No tendran validez impresiones de pantalla, correos electronicos de sus tutores, cartas manuscritas de algun padre o tutor, etc.
- Los motivos meramente personales quedan cubiertos por el 20% de faltas que se conviene para el estudiante.

Alumnos con Empleo (1)

Alumnos VIPs

En caso de tener un empleo formal dentro o fuera de la ciudad, es necesario entregar una **constancia laboral** que acredite el horario que se esta cubriendo (en el caso de locales, este horario se debe empalmar con el de la materia). En esa constancia debe acreditar que se esta haciendo labores de manera presencial en tal ubicacion. Esto lo dispensa solo del requisito de las asistencias, mas no de los proyectos que deban entregarse. Incluso pudiera solicitarle presentar avance de manera “remota” durante alguna de las clases. Enviar esa constancia con copia para el director de carrera.

Alumnos con Empleo (2)

- La justificación de inasistencias por *actividad laboral* se considerará a partir del momento de la recepción de dicha constancia en el correo del instructor (y no a partir de la fecha indicada en la constancia), por lo que si se recibe de manera tardía (con mas de una semana de retardo), dichas inasistencias NO SERAN justificadas.
- La justificación será válida si el estudiante programa **POR LO MENOS** dos asesorías por semana. De no hacerlo, pierde el beneficio de la justificación y se aplican las reglas anteriormente establecidas.

- 1 Introducción al cómputo móvil
 - 1 Fundamentos de Programación móvil
 - 2 Tipos de datos y expresiones
 - 3 Entornos de desarrollo de aplicaciones móviles
 - 4 Estructura de proyectos móviles
- 2 Diseño de Aplicaciones Móviles
 - 1 Interfaz de usuario
 - 2 Desarrollo de Aplicaciones Móviles
 - 3 Servicios y Notificaciones en Aplicaciones Móviles
- 3 Empleo de sensores en dispositivos móviles
 - 1 Gestión de Sensores
 - 2 Tópicos selectos de programación móvil

Evaluación (1)

- Para cada unidad del curso, se consideran 3 aspectos:
 - Ejercicios o investigaciones especiales (1)- 25%
 - Proyecto Individual - 35%
 - Proyecto en Equipo - 40%
- Para aprobar el curso, es obligatorio:
 - Tener calificación aprobatoria en todas las unidades (100-100-40 no da calificación aprobatoria).
 - Tener por lo menos dos asesorías por semana (Registrarlas por semana, no 30 asesorías al final del cuatrimestre)
 - Cumplir con el 80% de asistencia mínimo, incluyendo aquellas inasistencias justificadas debidamente

Evaluación (2)

Para cada unidad, habrá sesiones de “teoría”, sesiones de seguimiento de proyectos y sesiones de esparcimiento

- En las sesiones de teoría, el profesor presentará uno o varios temas
- En las sesiones de seguimiento de proyectos, de manera aleatoria se nombrará al integrante de equipo individual o en equipo. En el caso de que un integrante individual no responda, se le bajarán 5 puntos a su calificación del proyecto
- En las sesiones de esparcimiento, se permitirá a los estudiantes trabajar en proyectos pendientes, pero se contabilizará la asistencia.

Evaluación (3)

Sesiones de Seguimiento de proyectos

- En el caso de que el integrante del equipo seleccionado aleatoriamente no responda satisfactoriamente lo cuestionado, se le bajaran 5 puntos a su calificación del proyecto a todos los integrantes del equipo
- En el caso de los proyectos en equipo, el integrante seleccionado es aleatorio. Si en una primera ronda le toco al integrante A, en una segunda ronda posiblemente le toque al integrante B

Evaluación (4)

Lo que se debe presentar en una sesion de seguimiento de proyectos

- En un trabajo individual
 - Compartir pantalla de la ejecucion del avance del proyecto
 - Explicar con recursos multimedia los pasos para la resolucion del proyecto
 - Establecer el avance desde la ultima entrega
- En un trabajo grupal
 - Compartir pantalla de la ejecucion del avance del proyecto
 - Explicar con recursos multimedia los pasos para la resolucion del proyecto
 - Desglosar como se repartio el trabajo entre los integrantes del equipo
 - Establecer el avance desde la ultima entrega

Evaluación (5)

Acerca de los proyectos

- Aleatorios y DIFERENTES para la mayoría (preferentemente para cada integrante)
- Equipos: Proyectos diferentes para cada equipo, e Integrantes de los mismos formados de manera ALEATORIA!!

Fragmentación de equipos

- Si llegar a ocurrir que en un proyecto en equipo no hay un acuerdo para trabajar en equipo (Hay dos o mas entregas del proyecto asignado por partes diferentes dentro del mismo equipo)

Penalización

Cada “fragmento” de equipo recibe una penalizacion de 25 puntos mas las penalizaciones acumuladas por otros rubros.

- esta regla **NO APLICA** cuando hay uno o varios “desertores” del equipo (y hay una sola entrega del proyectos en equipo)

Acerca de Exención

- Cuando el profesor realiza alguna mecánica para exentar un proyecto (Individual/Equipo/Asignación especial) y uno o varios estudiantes completan lo solicitado, existen dos posibilidades:
 - El estudiante acepta exentar la elaboración de dicho proyecto o actividad, pero al hacer esto asume que la calificación asignada es 70.
 - El estudiante decide hacer el proyecto a pesar de haber exentado. En este caso el estudiante se hace acreedor a 20 puntos que puede aplicar sobre la calificación de dicho proyecto.

Cartucho de Recuperación (REC)

- Estudiante tiene derecho a solicitar un ÚNICO proyecto de recuperación aplicable a un solo proyecto o actividad.
- Esta solicitud debe HACERLA es estudiante - El profesor NO ES RESPONSABLE de informar al estudiante cuando tiene un ADEUDO.
- Si el proyecto no entregado es individual, se asigna otro proyecto diferente.
- Si el proyecto es en equipo, de común acuerdo con los integrantes pueden trabajar en otro proyecto diferente en equipo, o recibir una asignación individual de un proyecto diferente.
- La calificación recuperada será asignada siempre y cuando cumpla con el porcentaje de falta mínimo necesario para aprobar. Además, debe haber agendado el % de asesorías proporcional al tiempo de cuatrimestre transcurrido.
- El nuevo proyecto asignado esta diseñado para que el estudiante invierta en él por lo menos 1 SEMANA. Si lo solicita un día antes de terminar el cuatrimestre, posiblemente no tendrá tiempo de llevarlo a cabo.

Reporte Técnico de Desarrollo de Práctica

- Para cada práctica realizada, entregar un documento (**únicamente en formato PDF***) con las siguientes secciones:
 - Introducción
 - Desarrollo Experimental
 - Resultados
 - Conclusiones
 - **Referencias**
- Para GENERAR este reporte es necesario utilizar la plantilla en LATEX (**únicamente usando LATEX***) localizada en el siguiente enlace:
<https://www.overleaf.com/read/dgkhvfwnygvc>

Reporte Técnico de Desarrollo de Práctica

- Bajo ninguna circunstancia deben incluir **CÓDIGO FUENTE**. Si pueden incluir diagrama de flujo, Pseudocódigo, Diagrama E-R, Diagrama de Clases, de Casos de USO, etc. De incluir código fuente, solo tendrá un 50% del valor en la calificación.
- En caso de trabajos individuales o en EQUIPO, deben emplear la plantilla LaTeX que se provee. En caso de utilizar algo diferente a LaTeX u otra plantilla de LaTeX, la calificación proporcional del informe será **DESESTIMADA**.
- En caso de trabajos en equipo, se debe agregar los integrantes al inicio del INFORME. **El trabajo solo cuenta para aquellos integrantes mencionados en el informe (y que dicho nombre se encuentre registrado tal cual en la lista). Una vez ENTREGADO, si hay OMISIONES de los integrantes, no se realizará CORRECCION alguna, se debe asumir la consecuencias que esto conlleva.**

Ponderación del Informe en la Calificación del Proyecto

- Informe: 34 Puntos
 - Uso adecuado de Latex: 5 Puntos
 - Organización y Redacción: 6 Puntos
 - Referencias en formato adecuado: 8 Puntos
 - Evidencia del trabajo realizado: 8 Puntos
 - Sin faltas de ortografía ni errores de dedo: 7 Puntos
- Proyecto: 66 Puntos
 - Ejecución y Funcionalidad: 45 Puntos
 - Modularidad: 13 Puntos
 - Documentación: 8 Puntos

Entregables de proyecto individual (1)

- Sustituir **iti-0000** por la clave de grupo (ver su horario)
- Crear un archivo ZIP con el siguiente formato de nombre:
 - **iti-000000_uX_nuno_maganda_marco_aurelio**
- Dentro, debe contener lo siguiente:
 - **iti-000000_uX_nuno_maganda_marco_aurelio_source** (Carpeta con código fuente de la aplicación)
 - **iti-000000_uX_nuno_maganda_marco_aurelio_latex** (Carpeta con código fuente del informe)
 - **iti-000000_uX_nuno_maganda_marco_aurelio.apk** (Instalable (solo si se trata de una aplicación móvil))
 - **iti-000000_uX_nuno_maganda_marco_aurelio.pdf** (Informe)
- Donde:
 - **X** es el número de unidad a un dígito (1, 2, etc)
 - **Sustituir con sus apellidos y nombres de manera apropiada**
- NO DEBE HABER otros archivos .ZIP dentro del ZIP PRINCIPAL

Entregables de proyecto individual (2)

- En el caso que un proyecto individual sea asignado en equipo a varios estudiantes, el archivo entregable DEBE MANEJARSE como la de un proyecto individual
 - Solo un integrante del equipo carga en la plataforma el entregable individual.
 - El informe debe llevar los nombres de los integrantes del equipo que trabajaron (Si se omite a alguien, se asume que no trabajo en el proyecto).
 - NO ES NECESARIO que los otros integrantes marquen en el sistema la tarea como entregada, ya que se conoce su situación desde que se asigna el proyecto. El profesor ya sabe que ustedes van en equipo con el estudiante que hizo la entrega, y por eso deben asegurarse que en el informe entregado, vayan anotados sus nombres.

Entregables de proyectos en equipo

- Sustituir **iti-0000** por la clave de grupo (ver su horario)
- Crear un archivo ZIP con el siguiente formato de nombre:
 - **iti-000000_eq_NN_uX**
- Dentro, debe contener lo siguiente:
 - **iti-000000_eq_NN_uX_source** (Carpeta con código fuente de la aplicación)
 - **iti-000000_eq_NN_uX_latex** (Carpeta con código fuente del informe)
 - **iti-000000_eq_NN_uX.apk** (Instalable - Solo aplicaciones móviles)
 - **iti-000000_eq_NN_uX.pdf** (Informe)

Donde:

- **NN** es el número de equipo a dos dígitos (01, 02, etc)
- **X** es el número de unidad a un dígito (1, 2, etc)
- En cada entrega, **UN SOLO INTEGRANTE DEL EQUIPO** deberá cargar los archivos en el classroom.
- NO DEBE HABER otros archivos .ZIP dentro del ZIP PRINCIPAL

Entregables de asignaciones especiales

- Sustituir **iti-0000** por la clave de grupo (ver su horario)
- Crear un archivo ZIP con el siguiente formato de nombre:
 - **iti-000000_aeX_uY_nuno_maganda_marco_aurelio**
- Dentro, debe contener lo siguiente:
 - **iti-000000_aeX_uY_nuno_maganda_marco_aurelio_source** (Carpeta con código fuente de la aplicación - Cuando aplique)
 - **iti-000000_aeX_uY_nuno_maganda_marco_aurelio_latex** (Carpeta con código fuente del informe o diapositivas)
 - **iti-000000_aeX_uY_nuno_maganda_marco_aurelio.apk** (Instalable - Solo aplicaciones móviles, Cuando aplique)
 - **iti-000000_aeX_uY_nuno_maganda_marco_aurelio.pdf** (Informe)
- Donde:
 - **X** es el número de asignación dentro de la unidad a un dígito (1, 2, etc)
 - **Y** es el número de unidad a un dígito (1, 2, etc)
 - **Sustituir con sus apellidos y nombres de manera apropiada**
- NO DEBE HABER otros archivos .ZIP dentro del ZIP PRINCIPAL

Nombres de Archivos Entregables

En el caso de nombres y apellidos acentuados, con diéresis o con virgulilla (~), sustituir de acuerdo con las siguientes reglas:

- Sustituir N/n por Ñ/ñ
- Sustituir A/a por Á/á
- Sustituir E/e por É/é
- Sustituir I/i por Í/í
- Sustituir O/o por Ó/ó
- Sustituir U/u por Ú/ú
- Sustituir U/u por Ü/ü

Penalizaciones por Entregas Incompletas

- Proyecto que no este entregado de acuerdo con las especificaciones, será penalizado. Dos escenarios posibles:
 - El proyecto puede revisarse (completo o con faltas al formato).
 - El proyecto NO puede revisarse (falta codigo fuente, informe, APK, no se compila por alguna falla, etc). En automático el proyecto queda descartado.

Se recomienda LEER con cuidado la sección de entregables de esta presentación. Las penalizaciones son acumulables.

Detalle	Puntos de penalización sobre calificación final
Nombre Archivo	8
Tipo de Archivo	8
Estructura de Directorios	8
Falta o Error en Script	8
Poner ZIPs dentro del ZIP	8
Incluir ejecutables (aplica solo cuando el lenguaje es C++)	8

Premio a la Compresión Lectora 2024

A pesar de las instrucciones en esta presentación, alguien va a hacer las cosas mal



Salon de la Fama de Entregas Completas e Incompletas

Nombre
 iti-271229_u1_trevino_gandarilla_jesus_david_source
 iti-271229_u1_trevino_gandarilla_jesus_david_latex
 iti-271229_u1_trevino_gandarilla_jesus_david.pdf

Nombre
 iti-271229_u1_coyoy_lopez_mario_source
 iti-271229_u1_coyoy_lopez_mario_latex
 iti_271229_u1_coyoy_lopez_mario.pdf

Nombre
 iti-271229_u1_olivares_rodriguez_brayan_source
 iti-271229_u1_olivares_rodriguez_brayan_latex
 iti-271229_u1_olivares_rodriguez_brayan.pdf

Nombre
 iti-271229_u1_martinez_herrera_jose_guadalupe_source
 REPORTE INDIVIDUAL U2.zip
 REPORTE_INDIVIDUAL_U2.pdf

Nombre
 iti-271154_u1_rodriguez_porras_alejandra_carolina_latex.zip
 iti-271154_u1_rodriguez_porras_alejandra_carolina_source.zip
 iti_271154_u1_rodriguez_porras_alejandra_carolina_latex .pdf

Nombre
 iti-271229_u1_parras_pecina_maria_fernanda_latex
 iti-271229_u1_parras_pecina_maria_fernanda_source

Respecto a las Asignaciones Especiales

Importante

Si en algún momento del curso, la asignación especial consiste en desarrollar una presentación de un tema explicado en un BLOG, capítulo de libro, tutorial, etc., el LENGUAJE en el que deben hacerse las DIAPOSITIVAS es el MISMO que en el que está explicado dicho BLOG (a menos que se establezca un lenguaje diferente de manera explícita en el momento de asignar dicha tarea).

Nombre de Aplicación (Al crear su proyecto en Android Studio)

- Para los proyectos individuales, el nombre de la aplicación debe tener el siguiente formato (en MAYUSCULAS):

Z_CLAVEGRUPO_UX_APAT_AMAT_NOMBRE

Donde:

- Z es literal. La razón es para que al instalar la aplicación en el dispositivo de prueba, quede al final del listado
- X es Número de unidad en un dígito
- CLAVEGRUPO es la clave de grupo
- APAT, AMAT, NOMBRE - Datos del estudiante (Guiones bajo sustituyendo los espacios en blanco)
- Ejemplos:
 - Z_ITI-271234_U1_NUNO_MAGANDA_MARCO_AURELIO
 - Z_ITI-284433_U3_GUZMAN_LOERA_JOAQUIN
- Si en un proyecto individual se hacen equipos, entonces el formato del nombre de la APP cambiaría a:
Z_CLAVEGRUPO_UX_IND_E<NUMEROEQUIPO>
- Reemplazar <NUMEROEQUIPO> por el número de equipo asignado a dos dígitos.

Nombre de Aplicación (Al crear su proyecto en Android Studio)

- Para los proyectos en equipo, el nombre de la aplicación debe tener el siguiente formato (en MAYUSCULAS):
`Z_CLAVEGRUPO_UX_E<NUMEROEQUIPO>`
 - Z es literal. La razón es para que al instalar la aplicación en el dispositivo de prueba, quede al final del listado
 - X es el número de unidad en un dígito
 - *CLAVEGRUPO* es la clave de grupo
 - `<NUMEROEQUIPO>` Va a dos dígitos: 01, 02, 03
 - Ejemplos:
 - `Z_ITI-271234_U1_E01`
 - `Z_ITI-284433_U3_E04`

Nombre de Aplicación (Al crear su proyecto en Android Studio)

- Para las aplicaciones en Asignacion Especial (En caso de que se les solicite entregar el APK y el codigo fuente) (EN MAYUSCULAS), el nombre de la aplicacion debe tener el siguiente formato:
`Z_AEX_APAT_AMAT_NOMBRE`
 - Z es literal. La razon es para que al instalar la aplicacion en el dispositivo de prueba, quede al final del listado
 - X en el numero de unidad en un digito
 - APAT, AMAT, NOMBRE - Datos del estudiante (Guiones bajo sustituyendo los espacios en blanco)
 - Si llegara a ser en equipo, utilizar la nomenclatura de un proyecto en equipo dejando el inicio de la APP sin cambios (Z_AEX)

Package de su Aplicación

- Para proyectos individuales:
upvictoria_sep_dic_2024.iti_271086.pi1u1.nuno_maganda
- Sustituir iti_271086 por su GRUPO
- Sustituir nuno_maganda por sus apellidos paterno y materno respectivamente
- Ajustar pi1uX para proyectos proyecto individuales (proyecto individual 1 unidad X=1,2,3, etc)
- Si el proyecto es individual, debe llevar los apellidos paterno y materno
- Para proyectos en equipo:
upvictoria.pm_sep_dic_2023.iti_271086.pg1uX_eqYY
- Sustituir iti_271086 por su GRUPO
- Ajustar pg1uX para proyectos en equipo (proyecto grupal 1 unidad X = 1,2,3, etc)
-
- donde YY es el numero del equipo a dos digitos
- Para asignaciones especiales:
upvictoria.pm_sep_dic_2023.iti_271086.ae1uX.nuno_maganda

Fechas importantes de entrega de proyectos (1)

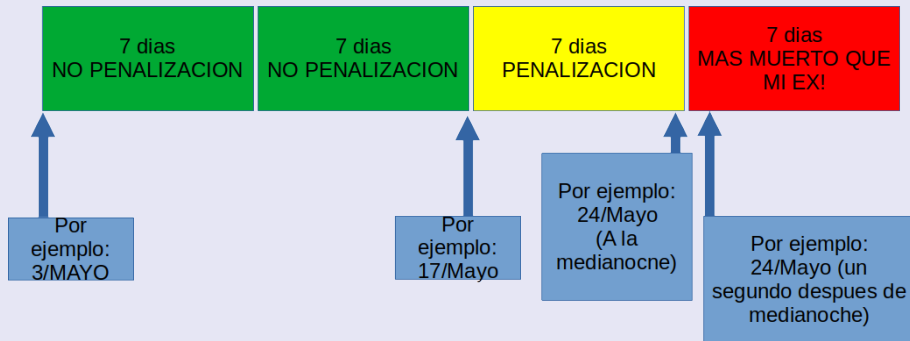
- Fecha de asignación: fecha en que se da a conocer al grupo el trabajo a elaborar
- Fecha de entrega sin penalización: 14 días naturales después de la fecha de asignación
- Proyecto entregado después de la fecha de penalización se le aplica una penalización de 25 PUNTOS
- Fecha de cierre: 21 días naturales después de la fecha de asignación.

Regla “CANTU”

- Ningún proyecto será revisado después de la fecha de cierre. Se programarán las entregas para cerrar y no permitir entregas tardías.

Fechas importantes de entrega de proyectos (2)

Grafo "DAFNE"



- En el momento de publicar la tarea, se incluirá la fecha para no penalización y fecha de cierre.

¿Es posible obtener una calificación negativa?

SI

Calificación asignada después de revisión	70
(-) Debería entregarse 17/Mayo, lo entregó 24/Mayo (1 minuto antes de la medianoche)	25
(-) Nombre Archivo MAL	8
(-) Tipo de Archivo MAL	8
(-) Estructura de Directorios INCORRECTA	8
(-) Faltas o Errores en Script	8
(-) ZIPs dentro del ZIP	8
(-) Incluir ejecutables (aplica solo cuando el lenguaje es C++)	8
(-) Penalización por Fragmentación de Equipo	25
Calificación Final	-28

LINUX

Recomendaciones

- No es obligatorio instalarlo, pero es recomendable por cuestión de desempeño.
- Si no quieren formatear computadora, se recomienda utilizar un HD booteable (SSD con persistencia) y bootear desde su laptop o computadora.
- Si lo instalan de manera nativa, puede ser cualquier distribución (**Mint, Ubuntu, Lubuntu, Xubuntu, Debian**).

Dispositivo Físico con Android

- Teléfono Inteligente/Tablet con Android Instalado (No afecta si no es la última versión)

Software Utilizado

Sobre una instalación de Linux, se debe instalar lo siguiente:

- Android Studio
- Scrcpy (<https://github.com/Genymobile/scrcpy>)
- Navegador Chrome/Firefox actualizado
- LaTeX para edición de reportes

Lenguajes Utilizados

Los lenguajes soportados por Android Studio son:

- Kotlin
- Java

Se hará énfasis en el lenguaje Kotlin, dado que es el lenguaje recomendado para nuevas aplicaciones, sin embargo, podría trabajarse también en Java dependiendo del proyecto

Se buscan integrantes para ingresar al
Salon de la fama del PLAGIO

Plagio

- Reprobación automática a quien reproduzca códigos de otros compañeros y los reporte como suyos, además de una nota en su expediente con copia para el consejo de calidad



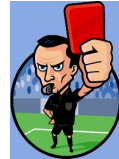
- Reprobación automática a quien copie códigos de Internet y los reporte como suyos, además de una nota en su expediente con copia para el consejo de calidad



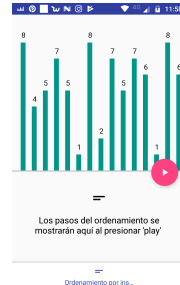
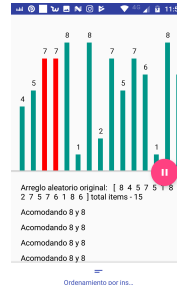
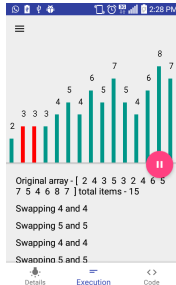
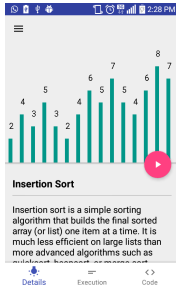
```

1 //Histogram equalization using C++ Image Processing
2 #include <iostream>
3 #include <opencv2/imgproc/imgproc.hpp>
4 #include <opencv2/highgui/highgui.hpp>
5 using namespace cv;
6 using namespace std;
7 using namespace cv;
8 using namespace cv;
9
10 //Funcion que inicializa todos los valores a 0
11 void initializeImage (Mat &image)
12 {
13     for (int i = 0; i < image.rows; i++)
14     {
15         for (int j = 0; j < image.cols; j++)
16             image.at<uchar>(i,j) = 0;
17     }
18 }
19
20 //Calculando el numero de pixeles de cada valor de intensidad
21 int *hist = new int[256];
22 for (int i = 0; i < image.rows; i++)
23     for (int j = 0; j < image.cols; j++)
24         hist[image.at<uchar>(i,j)]++;
25
26 //Funcion para calcular el histograma
27 void calculateHistogram (Mat &image, int *hist)
28 {
29     for (int i = 0; i < image.rows; i++)
30     {
31         for (int j = 0; j < image.cols; j++)
32             hist[image.at<uchar>(i,j)]++;
33     }
34 }
35
36 //Funcion que muestra los histogramas
37 void showHistograms (Mat &image, int *hist)
38 {
39     imshow("Histogram", hist);
40     waitKey(0);
41 }
42
43 //Calculando las histogramas
44 int *hist1 = new int[256];
45 for (int i = 0; i < image.rows; i++)
46     for (int j = 0; j < image.cols; j++)
47         hist1[image.at<uchar>(i,j)]++;
48
49 //Calculando las histogramas
50 int *hist2 = new int[256];
51 for (int i = 0; i < image.rows; i++)
52     for (int j = 0; j < image.cols; j++)
53         hist2[image.at<uchar>(i,j)]++;
54
55 //Normalizando la maxima intensidad del histograma
56 int max = hist1[0];
57 for (int i = 1; i < 256; i++)
58     if (hist1[i] > max)
59         max = hist1[i];
60
61 //Normalizando la maxima intensidad del histograma
62 int max2 = hist2[0];
63 for (int i = 1; i < 256; i++)
64     if (hist2[i] > max2)
65         max2 = hist2[i];
66
67 //Normalizando la maxima intensidad del histograma
68 int max3 = hist3[0];
69 for (int i = 1; i < 256; i++)
70     if (hist3[i] > max3)
71         max3 = hist3[i];
72
73 //Normalizando la maxima intensidad del histograma
74 int max4 = hist4[0];
75 for (int i = 1; i < 256; i++)
76     if (hist4[i] > max4)
77         max4 = hist4[i];
78
79 //Normalizando la maxima intensidad del histograma
80 int max5 = hist5[0];
81 for (int i = 1; i < 256; i++)
82     if (hist5[i] > max5)
83         max5 = hist5[i];
84
85 //Normalizando la maxima intensidad del histograma
86 int max6 = hist6[0];
87 for (int i = 1; i < 256; i++)
88     if (hist6[i] > max6)
89         max6 = hist6[i];
90
91 //Normalizando la maxima intensidad del histograma
92 int max7 = hist7[0];
93 for (int i = 1; i < 256; i++)
94     if (hist7[i] > max7)
95         max7 = hist7[i];
96
97 //Normalizando la maxima intensidad del histograma
98 int max8 = hist8[0];
99 for (int i = 1; i < 256; i++)
100     if (hist8[i] > max8)
101         max8 = hist8[i];
102
103 //Normalizando la maxima intensidad del histograma
104 int max9 = hist9[0];
105 for (int i = 1; i < 256; i++)
106     if (hist9[i] > max9)
107         max9 = hist9[i];
108
109 //Normalizando la maxima intensidad del histograma
110 int max10 = hist10[0];
111 for (int i = 1; i < 256; i++)
112     if (hist10[i] > max10)
113         max10 = hist10[i];
114
115 //Normalizando la maxima intensidad del histograma
116 int max11 = hist11[0];
117 for (int i = 1; i < 256; i++)
118     if (hist11[i] > max11)
119         max11 = hist11[i];
120
121 //Normalizando la maxima intensidad del histograma
122 int max12 = hist12[0];
123 for (int i = 1; i < 256; i++)
124     if (hist12[i] > max12)
125         max12 = hist12[i];
126
127 //Normalizando la maxima intensidad del histograma
128 int max13 = hist13[0];
129 for (int i = 1; i < 256; i++)
130     if (hist13[i] > max13)
131         max13 = hist13[i];
132
133 //Normalizando la maxima intensidad del histograma
134 int max14 = hist14[0];
135 for (int i = 1; i < 256; i++)
136     if (hist14[i] > max14)
137         max14 = hist14[i];
138
139 //Normalizando la maxima intensidad del histograma
140 int max15 = hist15[0];
141 for (int i = 1; i < 256; i++)
142     if (hist15[i] > max15)
143         max15 = hist15[i];
144
145 //Normalizando la maxima intensidad del histograma
146 int max16 = hist16[0];
147 for (int i = 1; i < 256; i++)
148     if (hist16[i] > max16)
149         max16 = hist16[i];
150
151 //Normalizando la maxima intensidad del histograma
152 int max17 = hist17[0];
153 for (int i = 1; i < 256; i++)
154     if (hist17[i] > max17)
155         max17 = hist17[i];
156
157 //Normalizando la maxima intensidad del histograma
158 int max18 = hist18[0];
159 for (int i = 1; i < 256; i++)
160     if (hist18[i] > max18)
161         max18 = hist18[i];
162
163 //Normalizando la maxima intensidad del histograma
164 int max19 = hist19[0];
165 for (int i = 1; i < 256; i++)
166     if (hist19[i] > max19)
167         max19 = hist19[i];
168
169 //Normalizando la maxima intensidad del histograma
170 int max20 = hist20[0];
171 for (int i = 1; i < 256; i++)
172     if (hist20[i] > max20)
173         max20 = hist20[i];
174
175 //Normalizando la maxima intensidad del histograma
176 int max21 = hist21[0];
177 for (int i = 1; i < 256; i++)
178     if (hist21[i] > max21)
179         max21 = hist21[i];
180
181 //Normalizando la maxima intensidad del histograma
182 int max22 = hist22[0];
183 for (int i = 1; i < 256; i++)
184     if (hist22[i] > max22)
185         max22 = hist22[i];
186
187 //Normalizando la maxima intensidad del histograma
188 int max23 = hist23[0];
189 for (int i = 1; i < 256; i++)
190     if (hist23[i] > max23)
191         max23 = hist23[i];
192
193 //Normalizando la maxima intensidad del histograma
194 int max24 = hist24[0];
195 for (int i = 1; i < 256; i++)
196     if (hist24[i] > max24)
197         max24 = hist24[i];
198
199 //Normalizando la maxima intensidad del histograma
200 int max25 = hist25[0];
201 for (int i = 1; i < 256; i++)
202     if (hist25[i] > max25)
203         max25 = hist25[i];
204
205 //Normalizando la maxima intensidad del histograma
206 int max26 = hist26[0];
207 for (int i = 1; i < 256; i++)
208     if (hist26[i] > max26)
209         max26 = hist26[i];
210
211 //Normalizando la maxima intensidad del histograma
212 int max27 = hist27[0];
213 for (int i = 1; i < 256; i++)
214     if (hist27[i] > max27)
215         max27 = hist27[i];
216
217 //Normalizando la maxima intensidad del histograma
218 int max28 = hist28[0];
219 for (int i = 1; i < 256; i++)
220     if (hist28[i] > max28)
221         max28 = hist28[i];
222
223 //Normalizando la maxima intensidad del histograma
224 int max29 = hist29[0];
225 for (int i = 1; i < 256; i++)
226     if (hist29[i] > max29)
227         max29 = hist29[i];
228
229 //Normalizando la maxima intensidad del histograma
230 int max30 = hist30[0];
231 for (int i = 1; i < 256; i++)
232     if (hist30[i] > max30)
233         max30 = hist30[i];
234
235 //Normalizando la maxima intensidad del histograma
236 int max31 = hist31[0];
237 for (int i = 1; i < 256; i++)
238     if (hist31[i] > max31)
239         max31 = hist31[i];
240
241 //Normalizando la maxima intensidad del histograma
242 int max32 = hist32[0];
243 for (int i = 1; i < 256; i++)
244     if (hist32[i] > max32)
245         max32 = hist32[i];
246
247 //Normalizando la maxima intensidad del histograma
248 int max33 = hist33[0];
249 for (int i = 1; i < 256; i++)
250     if (hist33[i] > max33)
251         max33 = hist33[i];
252
253 //Normalizando la maxima intensidad del histograma
254 int max34 = hist34[0];
255 for (int i = 1; i < 256; i++)
256     if (hist34[i] > max34)
257         max34 = hist34[i];
258
259 //Normalizando la maxima intensidad del histograma
260 int max35 = hist35[0];
261 for (int i = 1; i < 256; i++)
262     if (hist35[i] > max35)
263         max35 = hist35[i];
264
265 //Normalizando la maxima intensidad del histograma
266 int max36 = hist36[0];
267 for (int i = 1; i < 256; i++)
268     if (hist36[i] > max36)
269         max36 = hist36[i];
270
271 //Normalizando la maxima intensidad del histograma
272 int max37 = hist37[0];
273 for (int i = 1; i < 256; i++)
274     if (hist37[i] > max37)
275         max37 = hist37[i];
276
277 //Normalizando la maxima intensidad del histograma
278 int max38 = hist38[0];
279 for (int i = 1; i < 256; i++)
280     if (hist38[i] > max38)
281         max38 = hist38[i];
282
283 //Normalizando la maxima intensidad del histograma
284 int max39 = hist39[0];
285 for (int i = 1; i < 256; i++)
286     if (hist39[i] > max39)
287         max39 = hist39[i];
288
289 //Normalizando la maxima intensidad del histograma
290 int max40 = hist40[0];
291 for (int i = 1; i < 256; i++)
292     if (hist40[i] > max40)
293         max40 = hist40[i];
294
295 //Normalizando la maxima intensidad del histograma
296 int max41 = hist41[0];
297 for (int i = 1; i < 256; i++)
298     if (hist41[i] > max41)
299         max41 = hist41[i];
300
301 //Normalizando la maxima intensidad del histograma
302 int max42 = hist42[0];
303 for (int i = 1; i < 256; i++)
304     if (hist42[i] > max42)
305         max42 = hist42[i];
306
307 //Normalizando la maxima intensidad del histograma
308 int max43 = hist43[0];
309 for (int i = 1; i < 256; i++)
310     if (hist43[i] > max43)
311         max43 = hist43[i];
312
313 //Normalizando la maxima intensidad del histograma
314 int max44 = hist44[0];
315 for (int i = 1; i < 256; i++)
316     if (hist44[i] > max44)
317         max44 = hist44[i];
318
319 //Normalizando la maxima intensidad del histograma
320 int max45 = hist45[0];
321 for (int i = 1; i < 256; i++)
322     if (hist45[i] > max45)
323         max45 = hist45[i];
324
325 //Normalizando la maxima intensidad del histograma
326 int max46 = hist46[0];
327 for (int i = 1; i < 256; i++)
328     if (hist46[i] > max46)
329         max46 = hist46[i];
330
331 //Normalizando la maxima intensidad del histograma
332 int max47 = hist47[0];
333 for (int i = 1; i < 256; i++)
334     if (hist47[i] > max47)
335         max47 = hist47[i];
336
337 //Normalizando la maxima intensidad del histograma
338 int max48 = hist48[0];
339 for (int i = 1; i < 256; i++)
340     if (hist48[i] > max48)
341         max48 = hist48[i];
342
343 //Normalizando la maxima intensidad del histograma
344 int max49 = hist49[0];
345 for (int i = 1; i < 256; i++)
346     if (hist49[i] > max49)
347         max49 = hist49[i];
348
349 //Normalizando la maxima intensidad del histograma
350 int max50 = hist50[0];
351 for (int i = 1; i < 256; i++)
352     if (hist50[i] > max50)
353         max50 = hist50[i];
354
355 //Normalizando la maxima intensidad del histograma
356 int max51 = hist51[0];
357 for (int i = 1; i < 256; i++)
358     if (hist51[i] > max51)
359         max51 = hist51[i];
360
361 //Normalizando la maxima intensidad del histograma
362 int max52 = hist52[0];
363 for (int i = 1; i < 256; i++)
364     if (hist52[i] > max52)
365         max52 = hist52[i];
366
367 //Normalizando la maxima intensidad del histograma
368 int max53 = hist53[0];
369 for (int i = 1; i < 256; i++)
370     if (hist53[i] > max53)
371         max53 = hist53[i];
372
373 //Normalizando la maxima intensidad del histograma
374 int max54 = hist54[0];
375 for (int i = 1; i < 256; i++)
376     if (hist54[i] > max54)
377         max54 = hist54[i];
378
379 //Normalizando la maxima intensidad del histograma
380 int max55 = hist55[0];
381 for (int i = 1; i < 256; i++)
382     if (hist55[i] > max55)
383         max55 = hist55[i];
384
385 //Normalizando la maxima intensidad del histograma
386 int max56 = hist56[0];
387 for (int i = 1; i < 256; i++)
388     if (hist56[i] > max56)
389         max56 = hist56[i];
390
391 //Normalizando la maxima intensidad del histograma
392 int max57 = hist57[0];
393 for (int i = 1; i < 256; i++)
394     if (hist57[i] > max57)
395         max57 = hist57[i];
396
397 //Normalizando la maxima intensidad del histograma
398 int max58 = hist58[0];
399 for (int i = 1; i < 256; i++)
400     if (hist58[i] > max58)
401         max58 = hist58[i];
402
403 //Normalizando la maxima intensidad del histograma
404 int max59 = hist59[0];
405 for (int i = 1; i < 256; i++)
406     if (hist59[i] > max59)
407         max59 = hist59[i];
408
409 //Normalizando la maxima intensidad del histograma
410 int max60 = hist60[0];
411 for (int i = 1; i < 256; i++)
412     if (hist60[i] > max60)
413         max60 = hist60[i];
414
415 //Normalizando la maxima intensidad del histograma
416 int max61 = hist61[0];
417 for (int i = 1; i < 256; i++)
418     if (hist61[i] > max61)
419         max61 = hist61[i];
420
421 //Normalizando la maxima intensidad del histograma
422 int max62 = hist62[0];
423 for (int i = 1; i < 256; i++)
424     if (hist62[i] > max62)
425         max62 = hist62[i];
426
427 //Normalizando la maxima intensidad del histograma
428 int max63 = hist63[0];
429 for (int i = 1; i < 256; i++)
430     if (hist63[i] > max63)
431         max63 = hist63[i];
432
433 //Normalizando la maxima intensidad del histograma
434 int max64 = hist64[0];
435 for (int i = 1; i < 256; i++)
436     if (hist64[i] > max64)
437         max64 = hist64[i];
438
439 //Normalizando la maxima intensidad del histograma
440 int max65 = hist65[0];
441 for (int i = 1; i < 256; i++)
442     if (hist65[i] > max65)
443         max65 = hist65[i];
444
445 //Normalizando la maxima intensidad del histograma
446 int max66 = hist66[0];
447 for (int i = 1; i < 256; i++)
448     if (hist66[i] > max66)
449         max66 = hist66[i];
450
451 //Normalizando la maxima intensidad del histograma
452 int max67 = hist67[0];
453 for (int i = 1; i < 256; i++)
454     if (hist67[i] > max67)
455         max67 = hist67[i];
456
457 //Normalizando la maxima intensidad del histograma
458 int max68 = hist68[0];
459 for (int i = 1; i < 256; i++)
460     if (hist68[i] > max68)
461         max68 = hist68[i];
462
463 //Normalizando la maxima intensidad del histograma
464 int max69 = hist69[0];
465 for (int i = 1; i < 256; i++)
466     if (hist69[i] > max69)
467         max69 = hist69[i];
468
469 //Normalizando la maxima intensidad del histograma
470 int max70 = hist70[0];
471 for (int i = 1; i < 256; i++)
472     if (hist70[i] > max70)
473         max70 = hist70[i];
474
475 //Normalizando la maxima intensidad del histograma
476 int max71 = hist71[0];
477 for (int i = 1; i < 256; i++)
478     if (hist71[i] > max71)
479         max71 = hist71[i];
480
481 //Normalizando la maxima intensidad del histograma
482 int max72 = hist72[0];
483 for (int i = 1; i < 256; i++)
484     if (hist72[i] > max72)
485         max72 = hist72[i];
486
487 //Normalizando la maxima intensidad del histograma
488 int max73 = hist73[0];
489 for (int i = 1; i < 256; i++)
490     if (hist73[i] > max73)
491         max73 = hist73[i];
492
493 //Normalizando la maxima intensidad del histograma
494 int max74 = hist74[0];
495 for (int i = 1; i < 256; i++)
496     if (hist74[i] > max74)
497         max74 = hist74[i];
498
499 //Normalizando la maxima intensidad del histograma
500 int max75 = hist75[0];
501 for (int i = 1; i < 256; i++)
502     if (hist75[i] > max75)
503         max75 = hist75[i];
504
505 //Normalizando la maxima intensidad del histograma
506 int max76 = hist76[0];
507 for (int i = 1; i < 256; i++)
508     if (hist76[i] > max76)
509         max76 = hist76[i];
510
511 //Normalizando la maxima intensidad del histograma
512 int max77 = hist77[0];
513 for (int i = 1; i < 256; i++)
514     if (hist77[i] > max77)
515         max77 = hist77[i];
516
517 //Normalizando la maxima intensidad del histograma
518 int max78 = hist78[0];
519 for (int i = 1; i < 256; i++)
520     if (hist78[i] > max78)
521         max78 = hist78[i];
522
523 //Normalizando la maxima intensidad del histograma
524 int max79 = hist79[0];
525 for (int i = 1; i < 256; i++)
526     if (hist79[i] > max79)
527         max79 = hist79[i];
528
529 //Normalizando la maxima intensidad del histograma
530 int max80 = hist80[0];
531 for (int i = 1; i < 256; i++)
532     if (hist80[i] > max80)
533         max80 = hist80[i];
534
535 //Normalizando la maxima intensidad del histograma
536 int max81 = hist81[0];
537 for (int i = 1; i < 256; i++)
538     if (hist81[i] > max81)
539         max81 = hist81[i];
540
541 //Normalizando la maxima intensidad del histograma
542 int max82 = hist82[0];
543 for (int i = 1; i < 256; i++)
544     if (hist82[i] > max82)
545         max82 = hist82[i];
546
547 //Normalizando la maxima intensidad del histograma
548 int max83 = hist83[0];
549 for (int i = 1; i < 256; i++)
550     if (hist83[i] > max83)
551         max83 = hist83[i];
552
553 //Normalizando la maxima intensidad del histograma
554 int max84 = hist84[0];
555 for (int i = 1; i < 256; i++)
556     if (hist84[i] > max84)
557         max84 = hist84[i];
558
559 //Normalizando la maxima intensidad del histograma
560 int max85 = hist85[0];
561 for (int i = 1; i < 256; i++)
562     if (hist85[i] > max85)
563         max85 = hist85[i];
564
565 //Normalizando la maxima intensidad del histograma
566 int max86 = hist86[0];
567 for (int i = 1; i < 256; i++)
568     if (hist86[i] > max86)
569         max86 = hist86[i];
570
571 //Normalizando la maxima intensidad del histograma
572 int max87 = hist87[0];
573 for (int i = 1; i < 256; i++)
574     if (hist87[i] > max87)
575         max87 = hist87[i];
576
577 //Normalizando la maxima intensidad del histograma
578 int max88 = hist88[0];
579 for (int i = 1; i < 256; i++)
580     if (hist88[i] > max88)
581         max88 = hist88[i];
582
583 //Normalizando la maxima intensidad del histograma
584 int max89 = hist89[0];
585 for (int i = 1; i < 256; i++)
586     if (hist89[i] > max89)
587         max89 = hist89[i];
588
589 //Normalizando la maxima intensidad del histograma
590 int max90 = hist90[0];
591 for (int i = 1; i < 256; i++)
592     if (hist90[i] > max90)
593         max90 = hist90[i];
594
595 //Normalizando la maxima intensidad del histograma
596 int max91 = hist91[0];
597 for (int i = 1; i < 256; i++)
598     if (hist91[i] > max91)
599         max91 = hist91[i];
600
601 //Normalizando la maxima intensidad del histograma
602 int max92 = hist92[0];
603 for (int i = 1; i < 256; i++)
604     if (hist92[i] > max92)
605         max92 = hist92[i];
606
607 //Normalizando la maxima intensidad del histograma
608 int max93 = hist93[0];
609 for (int i = 1; i < 256; i++)
610     if (hist93[i] > max93)
611         max93 = hist93[i];
612
613 //Normalizando la maxima intensidad del histograma
614 int max94 = hist94[0];
615 for (int i = 1; i < 256; i++)
616     if (hist94[i] > max94)
617         max94 = hist94[i];
618
619 //Normalizando la maxima intensidad del histograma
620 int max95 = hist95[0];
621 for (int i = 1; i < 256; i++)
622     if (hist95[i] > max95)
623         max95 = hist95[i];
624
625 //Normalizando la maxima intensidad del histograma
626 int max96 = hist96[0];
627 for (int i = 1; i < 256; i++)
628     if (hist96[i] > max96)
629         max96 = hist96[i];
630
631 //Normalizando la maxima intensidad del histograma
632 int max97 = hist97[0];
633 for (int i = 1; i < 256; i++)
634     if (hist97[i] > max97)
635         max97 = hist97[i];
636
637 //Normalizando la maxima intensidad del histograma
638 int max98 = hist98[0];
639 for (int i = 1; i < 256; i++)
640     if (hist98[i] > max98)
641         max98 = hist98[i];
642
643 //Normalizando la maxima intensidad del histograma
644 int max99 = hist99[0];
645 for (int i = 1; i < 256; i++)
646     if (hist99[i] > max99)
647         max99 = hist99[i];
648
649 //Normalizando la maxima intensidad del histograma
650 int max100 = hist100[0];
651 for (int i = 1; i < 256; i++)
652     if (hist100[i] > max100)
653         max100 = hist100[i];
654
655 //Normalizando la maxima intensidad del histograma
656 int max101 = hist101[0];
657 for (int i = 1; i < 256; i++)
658     if (hist101[i] > max101)
659         max101 = hist101[i];
660
661 //Normalizando la maxima intensidad del histograma
662 int max102 = hist102[0];
663 for (int i = 1; i < 256; i++)
664     if (hist102[i] > max102)
665         max102 = hist102[i];
666
667 //Normalizando la maxima intensidad del histograma
668 int max103 = hist103[0];
669 for (int i = 1; i < 256; i++)
670     if (hist103[i] > max103)
671         max103 = hist103[i];
672
673 //Normalizando la maxima intensidad del histograma
674 int max104 = hist104[0];
675 for (int i = 1; i < 256; i++)
676     if (hist104[i] > max104)
677         max104 = hist104[i];
678
679 //Normalizando la maxima intensidad del histograma
680 int max105 = hist105[0];
681 for (int i = 1; i < 256; i++)
682     if (hist105[i] > max105)
683         max105 = hist105[i];
684
685 //Normalizando la maxima intensidad del histograma
686 int max106 = hist106[0];
687 for (int i = 1; i < 256; i++)
688     if (hist106[i] > max106)
689         max106 = hist106[i];
690
691 //Normalizando la maxima intensidad del histograma
692 int max107 = hist107[0];
693 for (int i = 1; i < 256; i++)
694     if (hist107[i] > max107)
695         max107 = hist107[i];
696
697 //Normalizando la maxima intensidad del histograma
698 int max108 = hist108[0];
699 for (int i = 1; i < 256; i++)
700     if (hist108[i] > max108)
701         max108 = hist108[i];
702
703 //Normalizando la maxima intensidad del histograma
704 int max109 = hist109[0];
705 for (int i = 1; i < 256; i++)
706     if (hist109[i] > max109)
707         max109 = hist109[i];
708
709 //Normalizando la maxima intensidad del histograma
710 int max110 = hist110[0];
711 for (int i = 1; i < 256; i++)
712     if (hist110[i] > max110)
713         max110 = hist110[i];
714
715 //Normalizando la maxima intensidad del histograma
716 int max111 = hist111[0];
717 for (int i = 1; i < 256; i++)
718     if (hist111[i] > max111)
719         max111 = hist111[i];
720
721 //Normalizando la maxima intensidad del histograma
722 int max112 = hist112[0];
723 for (int i = 1; i < 256; i++)
724     if (hist112[i] > max112)
725         max112 = hist112[i];
726
727 //Normalizando la maxima intensidad del histograma
728 int max113 = hist113[0];
729 for (int i = 1; i < 256; i++)
730     if (hist113[i] > max113)
731         max113 = hist113[i];
732
733 //Normalizando la maxima intensidad del histograma
734 int max114 = hist114[0];
735 for (int i = 1; i < 256; i++)
736     if (hist114[i] > max114)
737         max114 = hist114[i];
738
739 //Normalizando la maxima intensidad del histograma
740 int max115 = hist115[0];
741 for (int i = 1; i < 256; i++)
742     if (hist115[i] > max115)
743         max115 = hist115[i];
744
745 //Normalizando la maxima intensidad del histograma
746 int max116 = hist116[0];
747 for (int i = 1; i < 256; i++)
748     if (hist116[i] > max116)
749         max116 = hist116[i];
750
751 //Normalizando la maxima intensidad del histograma
752 int max117 = hist117[0];
753 for (int i = 1; i < 256; i++)
754     if (hist117[i] > max117)
755         max117 = hist117[i];
756
757 //Normalizando la maxima intensidad del histograma
758 int max118 = hist118[0];
759 for (int i = 1; i < 256; i++)
760     if (hist118[i] > max118)
761         max118 = hist118[i];
762
763 //Normalizando la maxima intensidad del histograma
764 int max119 = hist119[0];
765 for (int i = 1; i < 256; i++)
766     if (hist119[i] > max119)
767         max119 = hist119[i];
768
769 //Normalizando la maxima intensidad del histograma
770 int max120 = hist120[0];
771 for (int i = 1; i < 256; i++)
772     if (hist120[i] > max120)
773         max120 = hist120[i];
774
775 //Normalizando la maxima intensidad del histograma
776 int max121 = hist121[0];
777 for (int i = 1; i < 256; i++)
778     if (hist121[i] > max121)
779         max121 = hist121[i];
780
781 //Normalizando la maxima intensidad del histograma
782 int max122 = hist122[0];
783 for (int i = 1; i < 256; i++)
784     if (hist122[i] > max122)
785         max122 = hist122[i];
786
787 //Normalizando la maxima intensidad del histograma
788 int max123 = hist123[0];
789 for (int i = 1; i < 256; i++)
790     if (hist123[i] > max123)
791         max123 = hist123[i];
792
793 //Normalizando la maxima intensidad del histograma
794 int max124 = hist124[0];
795 for (int i = 1; i < 256; i++)
796     if (hist124[i] > max124)
797         max124 = hist124[i];
798
799 //Normalizando la maxima intensidad del histograma
800 int max125 = hist125[0];
801 for (int i = 1; i < 256; i++)
802     if (hist125[i] > max125)
803         max125 = hist125[i];
804
805 //Normalizando la maxima intensidad del histograma
806 int max126 = hist126[0];
807 for (int i = 1; i < 256; i++)
808     if (hist126[i] > max126)
809         max126 = hist126[i];
810
811 //Normalizando la maxima intensidad del histograma
812 int max127 = hist127[0];
813 for (int i = 1; i < 256; i++)
814     if (hist127[i] > max127)
815         max127 = hist127[i];
816
817 //Normalizando la maxima intensidad del histograma
818 int max128 = hist128[0];
819 for (int i = 1; i < 256; i++)
820     if (hist128[i] > max128)
821         max128 = hist128[i];
822
823 //Normalizando la maxima intensidad del histograma
824 int max129 = hist129[0];
825 for (int i = 1; i < 256; i++)
826     if (hist129[i] > max129)
827         max129 = hist129[i];
828
829 //Normalizando la maxima intensidad del histograma
830 int max130 = hist130[0];
831 for (int i = 1; i < 256; i++)
832     if (hist130[i] > max130)
833         max130 = hist130[i];
```

- Reprobación automática a quien copie códigos de Internet y los reporte como suyos, además de una nota en su expediente con copia para el consejo de calidad



<https://github.com/naman14/AlgorithmVisualizer-Android>



Frase célebre

“Finalmente son jóvenes que están en la preparatoria y que deben de leer su convocatoria con toda claridad, si no cumplen con los requisitos, si no pueden leer una convocatoria que dice tienes que traer número uno esto, número dos esto, número tres esto, no están listos para ser **estudiantes de educación superior**, así lo digo con toda claridad”.

Sara Ladrón de Guevara.

Rectora de la Universidad Veracruzana (2013-2017 y 2017-2021).

CONCLUSIÓN

