
	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115

## Tabla de Contenido

Motivación .....	2
Objetivo General.....	2
Objetivos Específicos .....	2
Antecedentes .....	2
Consideraciones generales .....	3
Metodología de revisión: .....	4
Entregables .....	4
Entregables parciales .....	4
Entregables finales: .....	5
Requerimientos del flujo de trabajo.....	6
¡Diseñemos un cajero automático! .....	8
Requerimientos Técnicos Mínimos .....	8
Estructura del Sistema de Archivos .....	19
Cronograma .....	24
Evaluación.....	25
Rubricas de evaluación tanto para los avances como entrega final.....	27
Evaluación interna del grupo para cada uno de los miembros .....	28
Directriz sobre Honestidad Académica.....	28
Bibliografía .....	30

	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115

## Motivación

En el proceso de formación de ingenieros de software es de suma importancia enfrentarse al diseño e implementación de programas complejos que exploten y demuestren la comprensión de las habilidades aprendidas. La escogencia de temáticas actuales y de uso cotidiano permite al estudiante comprender problemas computacionales de alto nivel de una forma más intuitiva y directa junto con una motivación intrínseca a poner en práctica el conocimiento adquirido.

## Objetivo General


Desarrollar un programa computacional en lenguaje Python que explote las habilidades de programación aprendidas.

## Objetivos Específicos

- Demostrar el entendimiento de los módulos solicitados a través de la generación de la identificación del problema, estructuras de datos requeridos y algoritmo necesario para la resolución del problema.
- Diseñar un programa que logre la cohesión de submódulos de una forma lógica para resolver los requerimientos de software.
- Defender el diseño implementado utilizando las correctas habilidades de comunicación para demostrar la resolución correcta del problema planteada.

## Antecedentes

Los cajeros automáticos se remontan a la década de 1930 cuando en 1939 se estrenó en New York, Estados Unidos los primeros prototipos. Sin embargo, no tuvieron éxito pues eran utilizados únicamente por pocas personas que querían evitar la interacción con los empleados bancarios y tenían confianza en dicho avance tecnológico. El primer cajero automático que tuvo éxito se remonta al año 1960, específicamente en Tokio, aunque existe poca información de su uso y éxito. El banco Barclays, en 1967, puso en ejercicio el primer cajero automático funcional después que John Shepherd-Barron, escocés trabajador de la empresa De La Rue (fabricante de máquinas de contar dinero), llegara tarde al banco y perder la oportunidad de sacar dinero. Se preguntó por qué debía

	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115


ser atendido por una persona para solicitar dinero y a partir de este hecho, nació el primer prototipo de cajero automático moderno. A la maquina diseñada se le conoce como **ATM** (*Automated Teller Machine*) o cajero automático en español. Además, se instauró el uso de cuatro dígitos como medio de autenticación universal (aunque este no es un requisito para su funcionamiento) a través de un **PIN** (*Personal Identification Number*).

Los cajeros automáticos hoy en día han cambiado el mundo y son parte de nuestro día a día. Aunque inicialmente estuvieron disponibles para personas pudientes, hoy son considerados un servicio básico más para cualquier agente activo de la sociedad [1] [2] [3].

Asimismo, se ha producido una evolución tecnológica de estos métodos de obtención de dinero en efectivo a tal punto que son en muchos casos elementos que sustituyen la interacción persona a persona para solicitar préstamos, hacer depósitos, pagar servicios, etc. Este tipo de productos adicionales son guiados por un codiseño y evolución del hardware y principalmente el software que han permitido una unificación de sistemas que producen una red global interconectada fiel reflejo del mundo globalizado en el que vivimos. El software se transforma día a día permitiendo avances en ciberseguridad, servicios, confiabilidad, autonomía, manejo de divisas y conversiones [4] [5].

## Consideraciones generales

1. El programa se debe desarrollar utilizando el lenguaje Python.
2. Es requerido utilizar todas las estructuras de software aprendidas y revisadas en la clase. Además, se deben cumplir con los requerimientos mínimos detallados en el programa del curso. Sin embargo, se pueden investigar e implementar cualquier estructura que se considere pertinente.
  - 2.1. Si el grupo de trabajo hace uso de alguna biblioteca y/o estructura no vista en clase debe primeramente consultar al profesor si su uso está correcta y permitido. Además, debe aparecer en la documentación una justificación del uso de dicha estructura/biblioteca.
3. Se debe seguir el estándar *Camel Case* para el nombre de los identificadores.

	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115

4. En semana 2 quedaran definidos los grupos de forma definitiva
  - 4.1. Ante algún conflicto el grupo debe buscar la mejor forma de resolverlo
  - 4.2. Ante abandono de algún estudiante del grupo, se debe buscar la forma de absorber las responsabilidades dejadas por el/la estudiante.

## Metodología de revisión:

Habrán tres avances y una presentación final (**ver cronograma y evaluación**).

1. Se dispondrá de una defensa oral de aproximadamente 20 minutos donde cada uno de los grupos deberá demostrar la resolución correcta del funcionamiento del programa implementado. Dicha defensa oral podrá incluir, pero no se limitará a:
  - 1.1. Demostración del funcionamiento lógico del programa
  - 1.2. Preguntas sobre estructuras implementadas (se podría solicitar que se abra un módulo específico como complemento a la respuesta dada por el estudiante)
2. El profesor hará una revisión del código fuente posterior a la defensa oral donde se evaluará la resolución correcta de las estructuras implementadas.
  - 2.1. Requerimientos mínimos de sistema para la revisión.
    - 2.1.1. Para dicha revisión se utilizará la versión de Python versión 3.10.2, instalado en Windows 10.
    - 2.1.2. El programa deberá ser ejecutado en una terminal de Windows.

## Entregables


### Entregables parciales

En cada avance se solicita se entregue una serie de módulos (código en Python) que demuestre el avance del grupo. Además, se solicita una documentación que tiene como principal objetivo que el estudiante identifique los elementos claves que le ayudaran a desarrollar cada módulo. Referirse al **cronograma** para más detalles.

En cada avance se requiere:

#### 1. Entrega grupal

- 1.1. Código fuente de los módulos solicitados
- 1.2. Archivo de texto y código fuente donde se incluyan detalles solicitados.

	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115

1.3. Entregar en el campus virtual un archivo de texto donde se incluya el URL del repositorio,

1.4. Tanto el código fuente como documentación debe estar disponible en el repositorio de cada grupo.

1.4.1. Solo se revisará lo reportado al repositorio

1.4.2. Ver **Requerimientos del flujo de trabajo** para mas detalles sobre el manejo del repositorio.

## 2. Entrega individual

2.1. Tabla de evaluación a sus compañeros. El profesor facilitara la plantilla. Dicha plantilla será un Excel donde se debe crear una pestaña con el nombre de cada uno de los integrantes del grupo.

2.2. Cabe destacar que dicha evaluación debe hacerse a conciencia y honestidad pues podrá afectar la nota de cada uno de sus compañeros.

2.3. El formato del nombre será:  
*ProyectoFinal\_Avance\_<numeroDeAvance>\_Grupo\_<numero(nombre)DeGrupo>\_EvaluacionIndividual\_<nombreDelEstudiante>.xlsx*

2.4. Este entregable tendrá un espacio en el campus virtual para que sea subido por cada estudiante.

## Entregables finales:

### Entrega grupal


1. Código fuente:

1.1. Se debe entregar todos los archivos necesarios para poder realizar una ejecución completa del programa.

2. Documentación:

2.1. Secciones mínimas requeridas:

- Portada
- Índice interactivo (debe tener hipervínculos a cada sección)
- Introducción
- Objetivos
- Descripción de los módulos implementados
  - Problema, estructuras de software requeridas y algoritmos.

	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115

- Se puede hacer uso de diagramas de flujo para solventar este requerimiento.
  - Requerimientos del Sistema
  - Bibliotecas/Estructuras complementarias (si aplica)
  - Manual de Usuario
    - Se deben detallar todas las instrucciones necesarias para que un usuario pueda hacer uso del programa.
    - Se puede hacer uso de pantallazos para demostrar el uso del programa.
  - Conclusiones
  - Bibliografía
3. Entregar en el campus virtual un archivo de texto donde se incluya el URL del repositorio,
  4. Tanto el código fuente como documentación debe estar disponible en el repositorio de cada grupo.
    - 4.1. Solo se revisará lo reportado al repositorio
    - 4.2. Ver **Requerimientos del flujo de trabajo** para más detalles sobre el manejo del repositorio.


### Entrega individual

1. Tabla de evaluación a sus compañeros. El profesor facilitara la plantilla. Dicha plantilla será un Excel donde se debe crear una pestaña con el nombre de cada uno de los integrantes del grupo.
2. Cabe destacar que dicha evaluación debe hacerse a conciencia y honestidad pues podrá afectar la nota de cada uno de sus compañeros.
3. El formato del nombre será:  
*ProyectoFinal\_EntregaFinal\_Grupo\_<numero(nombre)DeGrupo>\_EvaluacionIndividual\_<nombreDelEstudiante>.xlsx*

## Requerimientos del flujo de trabajo

Todos los requerimientos del proyecto del curso serán entregado a través de un repositorio Git siguiendo los siguientes requerimientos:

- 1- Abrir una cuenta Git en un *host* gratuito (se recomienda Bitbucket o GitHub)

	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115

2- Crear un repositorio siguiendo este formato:

<#deGrupo>\_ProgramacionBasicalC2023. Ejemplo:

Grupo01\_ProgramacionBasicalC2023

3- Si el repositorio se crea privado, deben dar acceso a **acm0993** (GitHub) o **acm\_0993** (Bitbucket).

4- El repositorio de Git debe contener dos *branches* (ramas): **master** y **develop**.

5- Inicialmente el branch de develop debe crearse desde master.

6- Cuando se este trabajando en el proyecto, el estudiante debe crear un branch nuevo desde develop y cuando el módulo esté terminado el branch debe fusionarse (*merge*) a develop. Cualquier corrección o modificación después del merge implicara una repetición del proceso (es decir, crear una nueva branch desde develop y merge después de los cambios). Una vez el código desarrollado en develop esté listo deberá fusionarse (merge) a master y una etiqueta (tag) debe ser creada. El proceso anterior se muestra gráficamente a continuación:

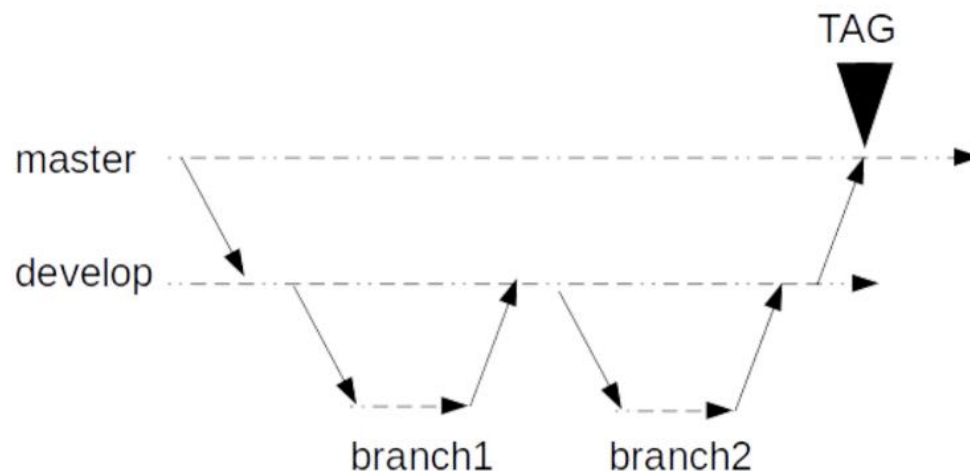



Figura 1. Muestra grafica del proceso que se debe seguir en el repositorio Git.

Cabe mencionar que se revisara únicamente lo que este contenido en el branch master posterior al tag.

7- Después de la realización de cada uno de los avances, la organización del repositorio debería verse similar a lo siguiente:

**Master** - ----- **Avance1**  
 |----- **Avance2**

	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115

|----- **Avance3**

|----- **EntregaFinal**

Cada directorio contendrá el entregable de cada asignación.

Se recomienda revisar los siguientes recursos en línea:

- <http://nvie.com/posts/a-successful-git-branching-model/>
- <https://rogerdudler.github.io/git-guide/index.es.html>

## ¡Diseñemos un cajero automático!

Ustedes como equipo de trabajo son contratados por la empresa Global Bank Inc. para presentar la arquitectura y diseño de un software de última generación para manejo de cajeros automáticos. Este software intenta recrear la idea inicial de los cajeros automáticos que será la sustitución de la interacción usuario/trabajador del banco. Por tanto, el cajero automático tendrá como parte de sus características el pago de servicios, depósitos y retiros de dinero en efectivo, registros de nuevos usuarios al sistema del banco, etc.

El gerente de la compañía le menciona que este sistema se espera este listo en un plazo máximo de 12 semanas.

## Requerimientos Técnicos Mínimos

### Menú Principal

Esto será lo primero que visualizará el usuario al momento de ejecutar el programa. Este modulo consta de cuatro opciones:


1. Registrar nuevo usuario
2. Usuario Registrado
3. Configuración Avanzada
4. Salir

Es esperado que cada una de las opciones (módulos) del menú anterior y cada uno de los menús que se explicaran a continuación correspondan a funciones independientes (en semana 8 se profundizara el tema de funciones).

### Registrar nuevo usuario

Este cajero tiene la posibilidad de gestionar el registro de usuarios que no estén registrados en el banco. Una vez que el usuario selecciona esta opción se desarrolla la siguiente secuencia:



	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115

#### 1. Solicitud de numero de cedula

- Debe ser de nueve dígitos.
  - En caso de que se digiten más de nueve dígitos se producirá un error y se debe alertar al usuario.
  - En caso de que se digiten menos de nueve dígitos se producirá un error y se debe alertar al usuario.
- No debe estar previamente registrado
  - El programa debe verificar que el numero de cedula ingresado no pertenezca a ningún usuario previamente registrado. En caso de que suceda se producirá un error y se debe alertar al usuario.
- Si sucede algún error según los criterios anteriores, el usuario tendrá hasta un máximo de tres intentos para ingresar una cedula valida.
  - Si consume los tres intentos, se volverá al menú principal y no se procederá con el registro del usuario.
  - Se debe desplegar un mensaje como por ejemplo: “Se excedió el máximo de intentos para ingresar un numero de cedula valido, volviendo al menú principal”

#### 2. Solicitud del nombre


- Se le permite al usuario registrar su nombre

#### 3. Escogencia del PIN

- El PIN es la forma en la que el usuario podrá acceder a su cuenta.
- Este PIN es de máximo 4 dígitos.
- El sistema le permitirá al usuario realizar la escogencia de su PIN y debe asegurarse que sea de 4 dígitos.
- Se le permitirán las veces que sea necesario hasta que el usuario digite un PIN valido.
- Una vez que el PIN es válido, se debe solicitar que se ingrese nuevamente para autenticar que es igual al PIN ingresado anteriormente. Si no es así, se debe advertir y volver a solicitarle que se ingrese nuevamente.
- El PIN es un autenticador único, por lo que no deben ser mostrado en pantalla pero el programa debe ser capaz de capturar la entrada.
  - Se sugiere el uso de la biblioteca *getpass* [6] [7]
    - Para prevenir la aparición de un ‘Warning’ en el *Shell* de Python se sugiere usar la terminal de su sistema operativo (CMD para Windows).

#### 4. Deposito obligatorio


- Para poder registrarse en el banco se debe realizar el depósito mínimo de 100 000 colones o el equivalente en alguna otra moneda (bitcoin o dólares)
  - Referirse a la explicación del proceso de depósito de dinero.

	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115

- En caso de depositar en una moneda que no sea colones, debe tomar en cuenta los tipos de cambio para verificar que el mínimo sea el equivalente a 100 000 colones.
    - El dinero depositado aparecerá en la cuenta asociada a la moneda seleccionada por el usuario.
      - Para las otras dos cuentas, se tendrá un valor inicial de cero.
  - El usuario tiene hasta un máximo de tres intentos para depositar el monto mínimo.
    - Si consume los tres intentos, se volverá al menú principal y no se procederá con el registro del usuario.
    - Se debe desplegar un mensaje como por ejemplo: “Se excedió el máximo de intentos para depositar el mínimo de dinero requerido, volviendo al menú principal”
5. [Este paso no será visible para el usuario] El sistema deberá seleccionar automática y aleatoriamente al menos tres servicios que se le asociaran al usuario.
- Además de la selección aleatoria, se le debe asignar a cada uno de los servicios el monto en colones a pagar y la condición activos.
  - Los servicios no seleccionados, se consideran desactivados.
  - Se sugiere el uso de la biblioteca *random* [8] [9].
6. [Este paso no será visible para el usuario] El sistema deberá guardar toda la información del nuevo usuario y generar el sistema de carpetas y archivos asociadas al usuario.
- Referirse a la explicación de la ‘Estructura del Sistema de Archivos’ para más detalles.
  - Se sugiere el uso de la biblioteca *os* [10] para la creación de directorios (carpetas) y lectura/escritura de archivos planos (en la semana 11 se profundizará con más detalle.)
7. Se regresa al menú principal


## Usuario Registrado

Este modulo se considera el corazón del programa debido a que es el encargado de darle al usuario la posibilidad de realizar trámites como depósitos y retiro de dinero,

	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115

pagos de servicios, eliminación de cuenta, entre otros. A continuación se muestra la dinámica esperada del programa:

1. Se debe cargar la información de los usuarios registrados desde un archivo de texto.
  - Referirse a la ‘Estructura del Sistema de Archivos’ para más detalles.
  - Se recomienda almacenar la información cargada desde los archivos en arreglos (más detalles sobre los arreglos en semana 9 y 10.)
2. El sistema deberá verificar que exista al menos un usuario registrado.
  - En caso de que no se cumpla este criterio, se debe alertar y volver al menú principal.
3. Autenticación de usuarios
  - El sistema solicitará el número de cédula del usuario.
    - El sistema deberá verificar que sea un número de cédula válido y registrado. En caso de que sea inválido:
      - Se le dará hasta un máximo de tres intentos para ingresar un número de cédula válido.
        - Si se agotan estos tres intentos, se devuelve al menú principal.
        - Se debe desplegar un mensaje como por ejemplo: “Se excedió el máximo de intentos para ingresar su cédula, volviendo al menú principal”
  - Si la cédula es válida, se le solicitará el PIN
    - Si se excede la cantidad máxima de tres intentos, se deberá volver al menú principal.
    - Se debe desplegar un mensaje como por ejemplo: “Se excedió el máximo de intentos para ingresar su PIN, volviendo al menú principal”
    - El PIN es un autenticador único, por lo que no deben ser mostrado en pantalla pero el programa debe ser capaz de capturar la entrada.
      - Se sugiere el uso de la biblioteca *getpass* [6] [7]

	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115

- Para prevenir la aparición de un '*Warning*' en el *Shell* de Python se sugiere usar la terminal de su sistema operativo (CMD para Windows).

4. Una vez se dio la autenticación correcta se desplegará un mensaje de bienvenida incluyendo el nombre con el que se registró el usuario. Cabe destacar que se debe cargar toda la información referente al usuario desde archivos de texto (saldos de los estados de cuenta e información de cada servicio).

- Referirse a la 'Estructura del Sistema de Archivos' para más detalles.
- Se recomienda almacenar la información en arreglos.

Además, se mostrará el submenú siguiente:

- Retirar dinero
- Depositar dinero
- Ver saldo actual
- Pagar servicios
- Compra/Venta de Divisas
- Eliminar usuario
- Salir

#### 4.1. Retirar Dinero

4.1.1. Debe consultar desde que cuenta (colones, dólares o bitcoin) desea hacer el retiro

4.1.1.1. Se sugiere que se despliegue la lista de cuentas disponibles y se le permita al usuario seleccionar a partir de las opciones, por ejemplo:


Cuentas disponibles:

1. Colones
2. Dólares
3. Bitcoin

¿De cual cuenta desea retirar dinero?

4.1.2. Una vez se seleccionó la cuenta, se debe preguntar al usuario cuanto quiere retirar

4.1.2.1. El sistema debe conocer cual es el saldo actual de las cuentas

	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115

4.1.3. Si el monto solicitado es mayor al saldo actual se debe advertir mostrando un mensaje y evitar la transacción.

4.1.3.1. Se le da hasta un máximo de tres oportunidades al usuario para digitar un monto valido.

4.1.3.1.1. Si falla mas de tres veces se devuelve al menú principal del sistema y se le advierte al usuario a través del despliegue de un mensaje.

4.1.3.1.1.1. Esto es una medida de seguridad debido a que el retiro incorrecto por mas de tres ocasiones podría significar un potencial fraude/robo de dinero.

4.1.3.1.1.2. Al salir al menú principal, debe asegurarse de actualizar los saldos (archivos) de cada una de las cuentas con el valor actual.

4.1.4. Si el monto solicitado es menor al saldo actual de la cuenta, se debe restar el monto solicitado e imprimir un mensaje indicando que se retiró el dinero solicitado y además, se debe imprimir el saldo actual.

4.1.5. Se regresa al submenú

## 4.2. Depositar dinero

4.2.1. Debe consultar a que cuenta (colones, dólares o bitcoin) desea hacer el deposito

4.2.1.1. Se sugiere que se despliegue la lista de cuentas disponibles y se le permita al usuario seleccionar a partir de las opciones, por ejemplo:

Cuentas disponibles:


4. Colones

5. Dólares

6. Bitcoin

¿A cuál cuenta desea acreditar el depósito de dinero?

4.2.2. Se debe preguntar al usuario cuanto quiere depositar. El monto debe ser positivo (requiere una verificación por parte del usuario y advertir en caso de error.)

	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115

4.2.3. Se imprime un mensaje indicando al usuario que la transacción se hizo de forma correcta y además se imprime el saldo anterior.

4.2.4. Se regresa al submenú

#### 4.3. Ver saldo Actual

4.3.1. Se imprime en pantalla el saldo actual de las tres cuentas disponibles.

4.3.2. Se regresa al submenú

#### 4.4. Pagar Servicios

4.4.1. Despliega un menú con cada uno de los servicios (Electricidad, Agua, Telefonía, Internet, Impuestos, Colegios Profesionales, Tarjeta de crédito).

4.4.2. Una vez seleccionado el servicio que desea cancelar, el sistema verifica si tiene este servicio activo.

4.4.2.1. Si no tiene, se regresa al menú de servicios

4.4.2.2. Si tiene, se procede al pago

4.4.2.3. Se muestra el saldo a pagar


4.4.2.4. Se consulta de cual cuenta (colones, dólares o bitcoin) desea hacer el debito

4.4.2.4.1. Se verifica que haya dinero suficiente de la cuenta seleccionada

4.4.2.4.1.1. El sistema debe realizar automáticamente la conversión de divisas en caso de que se escoja dólares o bitcoin.

4.4.2.4.2. Si hay dinero suficiente, se realiza el pago correspondiente y el rebajo del saldo de la cuenta seleccionada.

4.4.2.4.2.1. Si se escoge una divisa que no es colones, el rebajo de la cuenta debe hacerse en la moneda correspondiente. Por ejemplo: si se realiza el pago de 15 000 colones desde la cuenta en dólares y el banco vende el dólar en 550 colones quiere decir que necesitamos 27.27 dólares para pagar. Si se cuenta con saldo suficiente, se debe rebajar este monto (27.27 dólares) de la cuenta de dólares (ver 'Configuración Avanzada')

	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115

para saber de dónde se obtiene el valor de referencia de compra/venta de divisas).

4.4.2.5. Se regresa al submenú de servicios

#### 4.5. Compra/Venta de Divisas

4.5.1. El sistema consulta que desea realizar. Es decir, compra o venta de dólares, compra o venta de colones y compra o venta de bitcoin. Por ejemplo:  
¿Qué operación desea realizar?

1. Compra de colones
2. Venta de colones
3. Compra de dólares
4. Venta de dólares
5. Compra de bitcoin
6. Venta de bitcoin
7. Salir

4.5.2. Para la operación de compra se debe cumplir lo siguiente:


4.5.2.1. Se debe consultar la cuenta desde la que se obtendrá el dinero para la compra (cuenta origen). Por ejemplo: si se desean comprar dólares, se debe preguntar si el dinero se obtendrá desde la cuenta en colones o bitcoin.

Este proceso permitirá conocer el valor de conversión. Por ejemplo: si se selecciona bitcoin, se sabrá que se debe obtener el valor de compra de dólares vendiendo bitcoin (ver 'Configuración Avanzada' para saber de dónde se obtiene el valor de referencia de compra/venta de divisas).

4.5.2.2. Una vez seleccionada la cuenta del origen de los fondos, se debe consultar cuando se quiere comprar.

4.5.2.2.1. El sistema debe verificar que se cuente con el suficiente saldo para soportar la compra de la divisa.

4.5.2.2.1.1. Si se cuenta con el saldo suficiente, se realiza la operación y se deduce el monto de la cuenta origen y se acredita en la cuenta destino. Por ejemplo: si quiero comprar 1000 dólares desde mi cuenta de bitcoin y el precio de compra

	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115

es 1 bitcoin = 22 000 dólares, implica que yo debo tener al menos 0.04545 bitcoin en mi cuenta para poder realizar la compra. Al final de la transacción, si tengo solo 1 bitcoin y 0 dólares en mis respectivas cuentas, tendría 1000 dólares (cuenta destino) y 0.95455 bitcoin (cuenta origen).

4.5.2.2.1.2. Si no hay saldo suficiente en la cuenta origen, la operación se cancela y se vuelve al submenú anterior.

4.5.3. Para la operación de venta se debe cumplir lo siguiente:

4.5.3.1. Se debe consultar la cuenta donde se va a acreditar el dinero resultante de la venta (cuenta destino). Por ejemplo: si se desean vender dólares (cuenta origen), se debe preguntar si el dinero se acreditara en la cuenta en colones o bitcoin.

Este proceso permitirá conocer el valor de conversión. Por ejemplo: si se selecciona bitcoin, se sabrá que se debe obtener el valor de venta de dólares comprando bitcoin (ver 'Configuración Avanzada' para saber de dónde se obtiene el valor de referencia de compra/venta de divisas).


4.5.3.2. Una vez seleccionada la cuenta destino de los fondos, se debe consultar cuando se quiere vender.

4.5.3.2.1. El sistema debe verificar que se cuente con el suficiente saldo para soportar la venta de la divisa.

4.5.3.2.1.1. Si se cuenta con el saldo suficiente, se realiza la operación y se deduce el monto de la cuenta origen y se acredita en la cuenta destino. Por ejemplo: si quiero vender 1000 dólares desde mi cuenta de dólares y que se vean reflejados en bitcoin implica debería usar el precio de referencia para la operación 1 bitcoin = 0.00004348 dólares, implica que yo después de esta operación, obtendría 0.04348 bitcoin.

En mi saldo de bitcoin se me sumara ese valor obtenido y se me rebajaran los 1000 dólares de mi cuenta de dólares.



	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115

4.5.3.2.1.2. Si no hay saldo suficiente en la cuenta origen, la operación se cancela y se vuelve al submenú anterior.

4.5.4. Se volverá al submenú anterior.

#### 4.6. Eliminar Usuario

4.6.1. Esta operación eliminar todo registro del usuario. Esto implica que todas las carpetas y registros del usuario deberán ser borrados.

Se sugiere el uso de la biblioteca os [10]

4.6.2. Se le debe consultar el PIN al usuario.

4.6.2.1. Si coincide, se eliminan todos los registros (carpetas) asociadas al usuario.

Se sugiere el uso de la biblioteca os [8]

4.6.2.1.1. Se vuelve al menú principal

4.6.2.2. Si el PIN no coincide, se le advierte al usuario y se vuelve al submenú anterior.

4.6.3. El PIN es un autenticador único, por lo que no deben ser mostrado en pantalla pero el programa debe ser capaz de capturar la entrada.

4.6.3.1. Se sugiere el uso de la biblioteca *getpass* [6] [7]

4.6.3.1.1. Para prevenir la aparición de un '*Warning*' en el *Shell* de Python se sugiere usar la terminal de su sistema operativo (CMD para Windows).

#### 4.7. Salir


4.7.1. Se deben actualizar los archivos que contienen los saldos actuales y el estado de saldo de cada uno de los servicios.

4.7.2. Regresa al menú principal

#### 5. Configuración Avanzada

Aquí se pueden realizar modificaciones de los tipos de cambio y eliminación de usuarios.

5.1. Acceder a esta opción implica la autenticación como usuario avanzado. Esto implica la solicitud de un PIN especial.

	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115

5.1.1. Este PIN especial y sus características son definidas por el equipo de diseño. Es decir, deben definir cuestiones como longitud, sintaxis (numero, alfabético, alfanumérico, etc.)

5.1.2. En caso de que el usuario digite el PIN de forma errónea, se vuelve al menú principal.

5.1.3. El PIN es un autenticador único, por lo que no deben ser mostrado en pantalla pero el programa debe ser capaz de capturar la entrada.

5.1.3.1. Se sugiere el uso de la biblioteca *getpass* [6] [7]

5.1.3.1.1. Para prevenir la aparición de un '*Warning*' en el *Shell* de Python se sugiere usar la terminal de su sistema operativo (CMD para Windows).

5.2. Se despliega un menú:

- Eliminar usuario
- Modificar tipos de cambio
- Salir

5.2.1. Eliminar usuario

5.2.1.1. Se solicita el numero de cedula del usuario que se desea eliminar.

5.2.1.1.1. Si el usuario existe se procede a la eliminación de todos los registros (carpetas) asociadas al usuario.

Se sugiere el uso de la biblioteca *os* [10]


5.2.1.1.2. Si no existe, se advierte al usuario y se regresa al menú anterior.

5.2.2. Modificar tipos de cambio

5.2.2.1. Se despliega un menú que muestra cada uno de los tipos de conversión existentes, por ejemplo:

¿Qué tipo de cambio desea modificar?

1. Compra de colones
2. Venta de colones
3. Compra de dólares
4. Venta de dólares
5. Compra de bitcoin

	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115

6. Venta de bitcoin

7. Salir

5.2.2.2. Según el ítem seleccionado se procederá a realizar la modificación/actualización del archivo (s) que contiene cada factor de conversión. Lo que se espera como entrada de usuario es un numero que represente el factor de conversión. Los factores de conversión pueden ser inventados o ser reales a la situación macroeconómica nacional.

Los factores de conversión que deben existir son:

- De colon a bitcoin (venta de colones hacia cuenta bitcoin)
- De colon a dólar (venta de colones hacia cuenta dólares)
- De bitcoin a colones (venta de bitcoin hacia cuenta colones)
- De bitcoin a dólar (venta de colones hacia cuenta dólares)
- De dólares a colones (venta de dólares hacia cuenta colones)
- De dólares a bitcoin (venta de dólares hacia cuenta bitcoin)
- De colon a bitcoin (compra de colones desde cuenta bitcoin)
- De colon a dólar (compra de colones desde cuenta dólares)
- De bitcoin a colones (compra de bitcoin desde cuenta colones)
- De bitcoin a dólar (compra de colones desde cuenta dólares)
- De dólares a colones (compra de dólares desde cuenta colones)
- De dólares a bitcoin (compra de dólares desde cuenta bitcoin)

5.2.3. Salir

5.2.3.1. Regresa al menú principal

5.3. Salir


Se sale del programa sin ningún error.

## Estructura del Sistema de Archivos

El desarrollo de este programa tendrá un fuerte manejo de archivos y jerarquías. Es por esto por lo que se presenta un ejemplo de cómo se podría organizar el proyecto para tener éxito.

### Carpeta principal

La Figura 2 representa una sugerencia de cómo se podría ver la carpeta principal del proyecto. Esta carpeta incluye:

	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115

- Dos carpetas (123456789 y 987654321) que corresponden a los registros de dos usuarios y sus respectivos números de cedula. Cabe mencionar que en caso de que se elimine el usuario, se debe eliminar la carpeta asociada al número de cedula.
- El archivo '.py' que corresponde al código fuente.
- Un archivo llamado 'tipos\_de\_cambio.txt' (ver Figura 3). Este archivo contiene todos los tipos de cambio en el mismo orden expuesto en la sección 5.2.2.2.
- Un archivo llamado "usuarios\_pines.txt" (ver Figura 4). Este archivo contiene todos los pines y nombres de usuario (cedula) de cada uno de los usuarios registrados. Cabe mencionar que en caso que se elimine el usuario tanto sus numero de cedula y PIN se deben eliminar de esta lista. Además, la primera línea corresponde al PIN usado en la 'Configuración Avanzada'

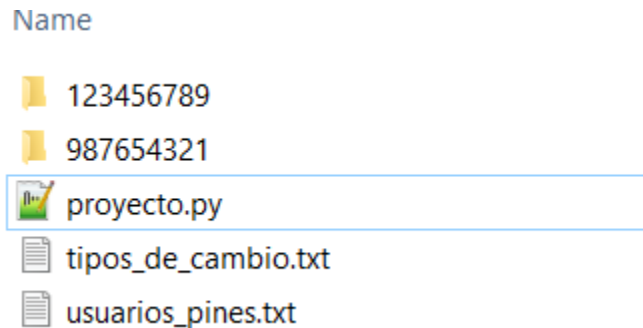


Figura 2. Sugerencia de organización de la carpeta principal

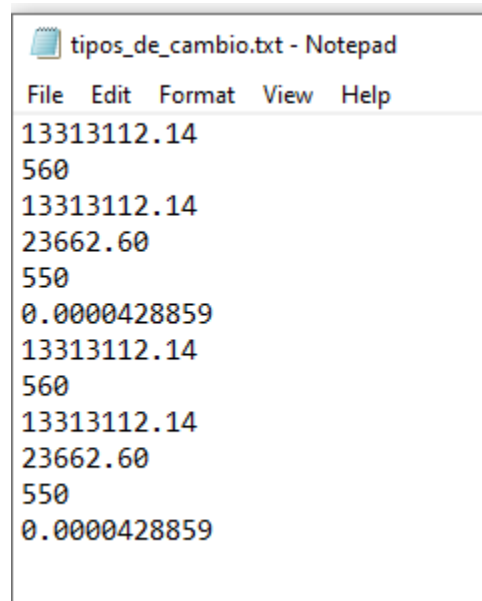

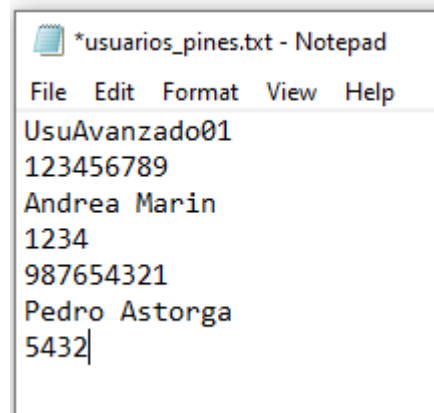


Figura 3. Ejemplo del archivo que contiene los tipos de cambio.

	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115



```

*usuarios_pines.txt - Notepad
File Edit Format View Help
UsuAvanzado01
123456789
Andrea Marin
1234
987654321
Pedro Astorga
5432

```

Figura 4. Ejemplo de archivo donde se contiene la información de cada usuario.

### Carpeta de cada usuario

En la carpeta de cada usuario, se sugiere tener al menos dos carpetas. Una para guardar la información de los saldos en las cuentas y otra para contener la información sobre el saldo adeudado de cada servicio y si esta activo. La Figura 5 muestra un ejemplo de como se sugiere su organización.

Se recuerda que esta carpeta debe ser eliminada si se 'Elimina el Usuario' desde el programa.

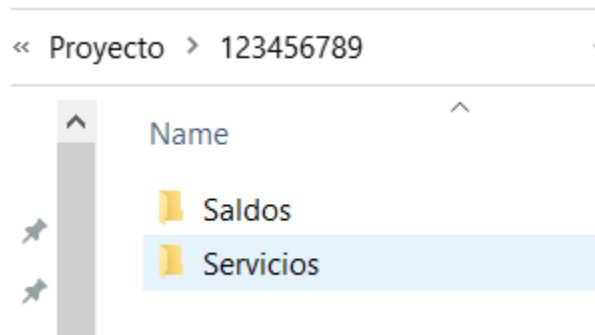



Figura 5. Ejemplo de organización de la carpeta de cada usuario

La Figura 6 muestra un ejemplo de la carpeta 'Saldos'. Esta debería contener un (varios) archivo(s) que contengan la información del saldo actual de cada cuenta. Este archivo debe actualizarse una vez que el usuario haya dejado de realizar operaciones bancarias (Usuario Registrado).

	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115

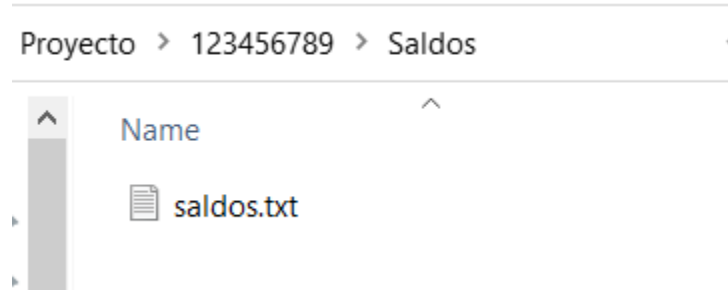


Figura 6. Ejemplo del contenido de la carpeta 'Saldos'

Además, en la Figura 7 se muestra un ejemplo del contenido del archivo 'saldos.txt'. Ejemplo: el usuario 123456789 tiene 100 000 colones, 0 dólares y 1 bitcoin.

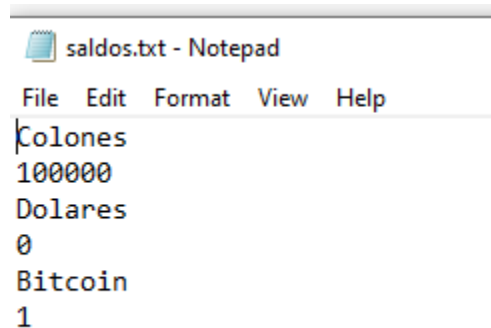


Figura 7. Ejemplo del contenido del archivo 'saldos.txt'

La Figura 8 muestra un ejemplo de lo que la carpeta 'Servicios' debería contener. Un archivo por cada uno de los 7 servicios disponibles para pago en el sistema.

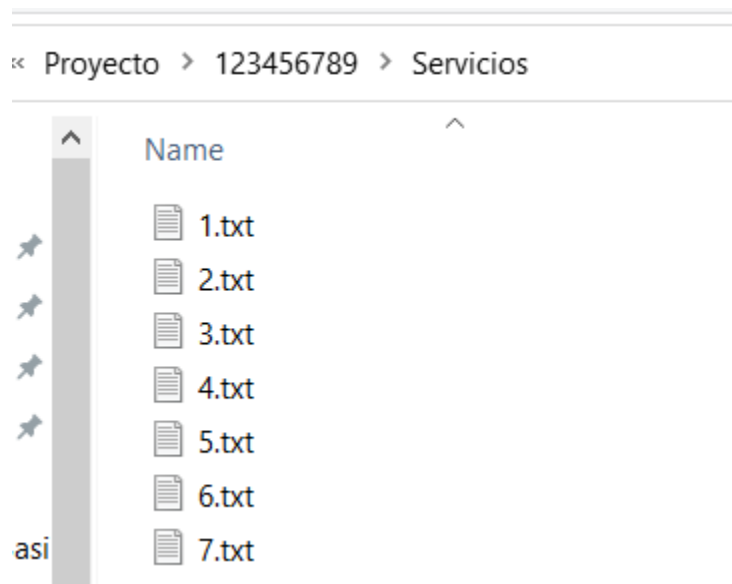



Figura 8. Ejemplo del contenido de la carpeta 'Servicios'

	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115

Además, la Figura 9 muestra un ejemplo de la estructura de cada uno de los archivos que contiene información sobre cada servicio. La primera línea contiene el nombre del servicio, la segunda línea si el servicio esta activo para este usuario y la tercera línea el monto a pagar.

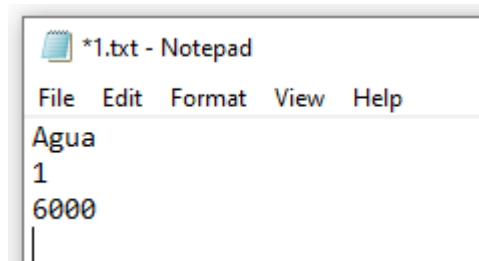



Figura 9. Ejemplo del contenido del archivo '1.txt'

#### Notas:

1. Las estructuras de datos utilizados para la resolución de cada requerimiento pueden ir en constante evolución a través del cuatrimestre. Es decir, estructuras que se utilizaron en el avance uno podría cambiar para cuando se dé la entrega final.
2. El estudiante puede investigar elementos de software adicionales a los explorados en la clase. Estos elementos de software adicionales deben ser debidamente enunciados y referenciados en la documentación. Además, se sugiere consultar al docente sobre estas estructuras ya que el uso de ellas puede aumentar la complejidad del proyecto (lo cual no es deseado).
3. Este enunciado debe cumplirse al 100%, alguna modificación y/o mejora de alguna característica debe consultarse primeramente con el profesor.
4. El uso de archivos es de suma importancia para el manejo y escalabilidad del proyecto. Es por esto por lo que se requiere que todos los datos iniciales como usuarios, PINes, saldos en cuentas, estado de los servicios y saldos adeudados deben mantenerse en archivos externos.
5. Dentro del modulo Usuarios Registrados, el saldo de una cuenta debe manejarse globalmente a través de una única variable. Esto para evitar saldos incorrectos. Por ejemplo: si el saldo inicial en colones es de 100 000 colones y

	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115


se retiran 5000, entonces el saldo que debe manejar el sistema para otras operaciones es 95 000 colones, no 100 000.

6. Si se observa, hay varias secciones del código que pueden ‘reciclarse’ en diferentes secciones por lo que se recomienda explotar el uso de funciones para escalar y facilitar el desarrollo y arquitectura del programa.

## Cronograma

SEMANA	VALOR	ENTREGABLES
Semana 8	0%	<ul style="list-style-type: none"> <li>Lectura del enunciado</li> <li>Para cada uno de los módulos y submódulos se debe entregar:               <ul style="list-style-type: none"> <li>Descripción general del problema</li> <li>Estructuras de Software (en este punto quizás no conozcan todas las estructuras por lo que esta parte estará en continua evolución)</li> <li>Algoritmo.</li> </ul> </li> <li>Código fuente escrito en lenguaje Python para               <ul style="list-style-type: none"> <li>Menú Principal y submenús (Con estructuras de decisión y ciclos)</li> <li>Modulo Registro Usuario (quizás no este terminado pero si se esperan avances)</li> </ul> </li> <li>Rubrica de autoevaluación de parte de cada uno de los miembros del grupo</li> <li>Manejo de Git: branches, commits, etc. Idealmente, lo revisado deberá estar el branch master.</li> </ul>
Semana 11	0%	<ul style="list-style-type: none"> <li>Código fuente escrito en lenguaje Python para:               <ul style="list-style-type: none"> <li>Modulo Registro Usuario (a este punto se espera que solamente guardar en los archivos este faltando)</li> <li>Usuarios Registrados (quizás no esté terminado pero si se esperan avances importantes)</li> </ul> </li> <li>Actualizaciones hechas a la descripción del problema, estructuras de software y algoritmo.</li> <li>Avances en la documentación</li> <li>Rubrica de autoevaluación de parte de cada uno de los miembros del grupo</li> <li>Manejo de Git: branches, commits, etc. Idealmente, lo revisado deberá estar el branch master.</li> </ul>
Semana 14	0%	<ul style="list-style-type: none"> <li>Código fuente escrito en lenguaje Python para:</li> </ul>




	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115

		<ul style="list-style-type: none"> <li>○ Modulo Registro Usuario (prácticamente terminado)</li> <li>○ Modulo Usuarios Registrados (prácticamente terminado)</li> <li>○ Modulo Configuraciones Avanzadas (prácticamente terminado)</li> <li>• Actualizaciones hechas a la descripción del problema, estructuras de software y algoritmo.</li> <li>• Avance de la documentación del proyecto (para este punto se espera que la documentación faltante sea manual de usuario y conclusiones)</li> <li>• Rubrica de autoevaluación de parte de cada uno de los miembros del grupo</li> <li>• Manejo de Git: branches, commits, etc. Idealmente, lo revisado deberá estar el branch master.</li> </ul>
Semana 15	<b>40%</b>	<ul style="list-style-type: none"> <li>• <i>Branch Master</i> en Git con la entrega del código fuente y documentación</li> <li>• Defensa del proyecto.</li> <li>• Rubrica de autoevaluación de parte de cada uno de los miembros del grupo</li> </ul>


## Evaluación

Proceso	Descripción	Tipo de evaluación	Valor
Defensa	Cada grupo tendrá aproximadamente 20 minutos para defender su implementación. Se debe ejecutar el programa y responder preguntas por parte del profesor. La evaluación es individual, ya que todos los miembros del grupo deben participar y demostrar su conocimiento sobre la implementación.	Individual	8
Código Fuente	Se evaluará que el código fuente conste de los requerimientos mínimos de calidad y que se cumpla con cada uno de los requerimientos. Esta evaluación la hará el profesor posterior a la defensa y se evaluará únicamente el código fuente entregado en el campus virtual.	Grupal*	8

	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115


Funcionamiento del Programa	Se evaluará que el código fuente entregado se logre ejecutar y cumpla de los requerimientos mínimos de calidad. Además, que se cumpla con cada uno de los requerimientos. Cabe destacar que se utilizara la documentación como referencia para instalar bibliotecas, considerar requerimientos mínimos, etc. Esta evaluación la hará el profesor posterior a la defensa y se evaluará únicamente el código fuente entregado en el campus virtual.	Grupal*	8
Control de versiones en Git	El entregable hace uso correcto del repositorio y cumple con la disposición sugerida y sigue el flujo de trabajo requerido	Grupal*	8
Documentación	Se evaluará que la documentación entregada cumpla con los capítulos mínimos estipulados en el enunciado junto con los estándares mínimos esperados de un documento de nivel universitario. Además, esta documentación será de suma importancia para comprender la implementación y ejecución de su código fuente. Esta evaluación la hará el profesor posterior a la defensa y se evaluará únicamente el código fuente entregado en el campus virtual.	Grupal*	8
<b>Total</b>			<b>40</b>

**\*La evaluación grupal contempla lo que cada integrante del grupo reporte en la evaluación individual de cada uno de los compañeros de grupo.**

	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115

## Rubricas de evaluación tanto para los avances como entrega final

Título: Programación Básica y sus aplicaciones para la computación, – Proyecto Final					
Producto: Informe y presentación del proyecto sobre la implementación de un proyecto complejo utilizando los temas vistos en el curso					
Concepto Rango	Excelente (100%)	Muy Bien (90%)	Bien (80%)	Regular (70%-50%)	No lo hace (0%)
Tiempo de entrega (10%)	El producto de la actividad se entregó en el día y la hora indicados.	El producto de la actividad se entregó hasta con medio día de retraso.	El producto de la actividad se entregó con un día de retraso.	El producto de la actividad se entregó hasta con dos días de atraso.	El producto de la actividad se entregó más de dos días tarde o no se entregó
Desarrollo del tema (70%)	Las ideas son ampliamente desarrolladas, son muy pertinentes al tema y aportan enfoques novedosos.	Las ideas son ampliamente desarrolladas y son pertinentes al tema.	Las ideas son medianamente desarrolladas y son pertinentes al tema.	Las ideas son pertinentes al tema, pero no están desarrolladas.	Las ideas desarrolladas no son pertinentes al tema ni significan un aporte al curso.
Presentación del trabajo (10%)	La forma del producto es la propuesta, la redacción y ortografía son excelentes.	La forma del producto es la propuesta, hay pocas faltas de redacción y ortografía.	La forma del producto es la propuesta, hay faltas de redacción y ortografía.	Faltan partes en la estructura del producto propuesta, la redacción es difícil de entender.	La estructura del producto no es la propuesta, no se entiende lo que dice el texto
Liderazgo y empatía (10%)	Participa en todas las reuniones del equipo, liderando con empatía y responsabilidad los aspectos del proyecto.	Participa en todas las reuniones del equipo, liderando con responsabilidad los aspectos del proyecto.	Participa en algunas de las reuniones del equipo, responsabiliza los aspectos del proyecto.	Participa en menos de 2 de las reuniones del equipo, no evidencia responsabilidad en los aspectos del proyecto.	No participa en las reuniones del equipo, nula responsabilidad en los aspectos del proyecto.
Total, Final:					

	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115

## Evaluación interna del grupo para cada uno de los miembros


El estudiante obtendrá el porcentaje de la nota grupal de la entrega de acuerdo con su participación, si participó en todo un 100% tendrá ese porcentaje de la entrega, si no, el que le sea asignado por su equipo. La tabla se entregará una por cada participante y será evaluada por todos los miembros del grupo.

RUBRO	EXCELENTE 25%	MUY BUENO 15%	BUENO 10%	DEFICIENTE 2%	NO REALIZADO 0 puntos
Asiste a todas las reuniones de coordinación propuestas por el equipo de trabajo					
Contribuye con la propuesta del trabajo que hay que desarrollar, dando ideas e investigando formas de lograr lo propuesto.					
Actúa constructivamente al afrontar cualquier conflicto o dificultad que se presente en el desarrollo del trabajo					
Entrega en tiempo y forma los trabajos asignados a su persona					
Total de porcentaje de participación					

## Directriz sobre Honestidad Académica

Para efectos de este proyecto, los participantes deben evitar conductas deshonestas tales como el fraude académico o plagio:

- Hacer fraude académico incluye, dentro de otras acciones, falsificar bibliografía, utilizar datos inventados, presentar como propios proyectos elaborados por otras


	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115

personas, obtener ayuda no autorizada en tareas calificadas o que otra persona desarrolle el trabajo que le corresponde a usted.

- Plagiar incluye copiar textualmente frases, oraciones, párrafos y trozos enteros de material impreso, Internet y otras fuentes, sin realizar la correspondiente cita; incluso parafrasear sin citar las fuentes.

Las situaciones anteriormente indicadas se penalizarán según el artículo 31 del reglamento estudiantil vigente, por lo que en una primera ocasión que se detecte y documente una falta el profesor consignará una nota de cero a la actividad evaluativa, y comunicará a vida estudiantil el hecho para su debido registro en el expediente académico del estudiante, si se detecta una segunda incidencia por parte del estudiante automáticamente pierde el curso y en una tercera ocasión documentada (independientemente del curso) provoca la pérdida de todos los cursos matriculados en ese cuatrimestre y la expulsión del programa académico y de la Universidad.

"Cree en ti mismo: Protagoniza tu vida. No te victimices. Cree en ti mismo con tanta fuerza, que el mundo no pueda evitar creer en ti también. Ten voluntad. Pon de tu parte. Merécete la felicidad. El premio más grande es la sensación de capacidad; para eso, necesitas poner tu mejor esfuerzo. Preocúpate por darte a los demás. A pesar de tu tragedia, otros necesitan de ti. Si aprendes a recibir, pronto sentirás la necesidad de dar. Sé objetivo. La vida no es dura, sólo hay momentos difíciles. Absolutamente todo es pasajero." (Hábitos de las Personas resilientes)

	<b>Ingeniería en Sistemas de Computación</b> Proyecto Final	Curso: Programación Básica
		Profesor: MSc. Alvaro Camacho Mora
		Código curso: SC-115

## Bibliografía

- [1] BBVA, «Historia de los Cajeros Automáticos,» 14 Julio 2020. [En línea]. Available: <https://www.bbva.com/es/historia-de-los-cajeros-automaticos/>. [Último acceso: 28 Enero 2023].
- [2] BBC Mundo, «La curiosa,» 2021 Junio 27. [En línea]. Available: <https://www.bbc.com/mundo/noticias-40417156>. [Último acceso: 28 Enero 2023].
- [3] Kuvasz, «Historia de los Cajeros Automáticos,» 08 Septiembre 2022. [En línea]. Available: <https://www.kvz.cl/historia-de-los-cajeros-automaticos/#:~:text=cheques%20en%20ella.-,El%20primer%20cajero%20autom%C3%A1tico%20se%20instal%C3%B3%20en%20junio%20de%201967,GBP%2010%20a%20la%20vez..> [Último acceso: 28 Enero 2023].
- [4] LinkedIn, «The Evolution of ATM Technology and the Future of Banking,» 20 Junio 2022. [En línea]. Available: [https://www.linkedin.com/pulse/evolution-atm-technology-future-banking-articles-network?trk=pulse-article\\_more-articles\\_related-content-card](https://www.linkedin.com/pulse/evolution-atm-technology-future-banking-articles-network?trk=pulse-article_more-articles_related-content-card). [Último acceso: 28 Enero 2023].
- [5] T. Muradzikwa, «ATM Innovation Over 50 Years Later,» 09 Diciembre 2022. [En línea]. Available: <https://www.plugandplaytechcenter.com/resources/atm-innovation/>. [Último acceso: 28 Enero 2023].
- [6] Python Documentation, «getpass — Portable password input,» 28 Enero 2023. [En línea]. Available: <https://docs.python.org/3/library/getpass.html>. [Último acceso: 28 Enero 2023].
- [7] GeeksforGeeks, «getpass() and getuser() in Python (Password without echo),» 15 Septiembre 2022. [En línea]. Available: <https://www.geeksforgeeks.org/getpass-and-getuser-in-python-password-without-echo/>. [Último acceso: 28 Enero 2023].
- [8] Python Documentation, «random — Generate pseudo-random numbers,» 28 Enero 2022. [En línea]. Available: <https://docs.python.org/3/library/random.html>. [Último acceso: 28 Enero 2022].
- [9] N. Aggarwal, «Python Random Module,» 14 Diciembre 2021. [En línea]. Available: <https://www.geeksforgeeks.org/python-random-module/>. [Último acceso: 28 Enero 2022].
- [10] Python Documentation, «os — Interfaces misceláneas del sistema operativo,» 28 Enero 2023. [En línea]. Available: <https://docs.python.org/es/3.10/library/os.html>. [Último acceso: 28 Enero 2023].
- [11] A. Pais, «Qué es la serie de Fibonacci y qué tiene que ver con el número áureo,» 24 Abril 2021. [En línea]. Available: <https://www.bbc.com/mundo/noticias-56691239>.