Teoria 1

1.-

Programa de aplicación: Firefox.exe

Sistema operativo: Windows

2.-

Coger el coche para ir al instituto:

1. Salir de casa
2. Ir hacia el coche
3. Abrir el coche
4. Entrar en el coche
5. Arrancar el coche
6. Conducir hacia el instituto
7. Respetar las normas de trafico
8. Aparcar cerca del destino
9. Parar el coche
10. Salir del coche
11. Cerrar el coche
12. Alejarse del coche

Cocinar patatas fritas:

1. Tener patatas
2. Pelar patatas
3. Poner el aceite a hervir
4. Cortar las patatas
5. Poner las patatas en el aceite hirviendo
6. Toque de sal, pimentón dulce y pimienta
7. Esperar un rato
8. Sacar las patatas del aceite
9. Secar las patatas con papel
10. Comer

3.-

1. De bajo nivel = Ensamblador
2. De alto nivel = Java
3. De propósito general = Pascal
4. De propósito específico = PHP
5. De programación de sistemas = Swift
6. De Script = Ruby
7. Compilado = Go
8. Interpretado = Python
9. Imperativo = Algol
10. Declarativo = Prolog
11. Procedural = Kotlin
12. Orientado a objetos = Visual Basic
13. De programación lógica = Python
14. De servidor = ASP.NET
15. De cliente = React

4.-

1. España
   1. Angulas
   2. React
   3. Django
   4. Ruby
   5. Laravel
   6. Spring
2. Mundo
   1. Python
   2. C
   3. Java
   4. C++
   5. C#
   6. JavaScript

5.-

1. **Análisis de Requerimientos y Modelo de Ciclo de Vida**:
   * Requerimientos del sistema:
     + Funcional
       1. Generar facturas de ventas.
          1. Entrada:

Cliente: DNI, Nombre, Apellidos, Dirección

Vendedor: Nºventa, Local\_venta, Fecha\_venta

* + - * 1. Salida:

Factura: Datos anteriores, id\_producto, Datos\_venta, Precio\_final

* + - 1. Registrar ventas por trabajador.
         1. Entrada:

Id\_trabajador, Nombre, Apellidos, id\_producto, precio\_final

* + - 1. Controlar el stock de productos.
         1. Entrada:

Id\_producto, nºvenidos

* + - * 1. Salida:

Id\_producto, nºexistencias

* + - 1. Integración con lector de código de barras y tarjetas de crédito.
         1. Entrada:

Codigo de barras

* + - * 1. Salida:

Id\_producto, precio\_producto

* + - 1. Administrar precios de productos y operar con ellos.
         1. Entrada:

Id\_producto, precio\_actualizado, nºexistencias

* + - * 1. Salida:

Datos actualizados

* + - 1. Almacenar información de trabajadores y productos.
         1. Entrada:

Id\_producto, precio\_prodcuto, nºexistencias

Id\_trabajador, Nombre, Apellidos, Salario

* + - No funcional
      1. Minimizar el tiempo de respuesta.
      2. Gestión de usuarios para evitar procesamiento simultáneo.
  + Diseño: Utilizaremos el modelo de ciclo de vida en espiral ya que al ser una empresa de ventas puede agregar productos y necesitar mas opciones y funcionalidades, este modelo nos permite replanificar e implementar nuevas extensiones al programa o modificar las ya existentes siempre que se necesite

1. **Planificación de la Codificación**:
   * Lenguaje de programación: Java.
   * Herramientas:
     + IDE: Eclipse o IntelliJ IDEA para desarrollo Java.
     + Gestión de versiones: Git para control de código fuente.
     + Base de datos: ApacheDerby para almacenar datos de trabajadores y productos.
     + Integración de lectores de código de barras y pagos con SDKs específicos.
   * Elección de herramientas: Java es versátil y ampliamente utilizado, permitiendo una fácil integración de bibliotecas para lectores de código de barras y tarjetas de crédito, también es el lenguaje utilizado en ApacheDerby cosa que facilita mucho la comunicación entre la aplicación y la base de datos
2. **Planificación de las Restantes Fases del Ciclo de Vida**:
   * **Diseño**:
     + Objetivo: Diseñar la arquitectura de la aplicación, esquemas de base de datos y definir las interfaces de usuario.
     + Cómo: Utilizar UML para modelar la arquitectura y diseñar las pantallas de la aplicación.
   * **Implementación**:
     + Objetivo: Codificar la aplicación siguiendo el diseño establecido.
     + Cómo: Desarrollar módulos para facturación, gestión de stock, autenticación de usuarios, y la integración de lectores y pagos.
   * **Pruebas**:
     + Objetivo: Verificar que la aplicación cumple con los requisitos y funciona correctamente.
     + Cómo: Realizar pruebas unitarias, de integración y de sistema para identificar y solucionar errores.
   * **Despliegue**:
     + Objetivo: Implementar la aplicación en el entorno de producción.
     + Cómo: Configurar un servidor web para alojar la aplicación y asegurar que esté disponible para su uso.
   * **Mantenimiento y Soporte**:
     + Objetivo: Mantener la aplicación, corregir errores y brindar soporte técnico.
     + Cómo: Establecer un proceso de seguimiento de problemas y actualizaciones periódicas.