**Pseudocódigo del Proyecto**

imprimir("Un saludo de paz y bien, bienvenidos\nLas opciones de calculo son las siguientes:\nSelecciones el numero de operación que desee.\n1)Conversor de longitud (1).\n2)Conversor de peso (2).\n3)Conversor de Bytes (3).\n4)Conversor de base numérica.(4)")

Repetir hasta que se cumpla:

    opcion = input("Ingresa un número del 1 al 4: ")

    Si opcion.Comprobar valor positivo y verdadero and integer(opcion) in range(1, 5):

        opcion = integer(opcion)

        Cerrar cuando se cumple el ciclo

    en caso (contrario) de:

        imprimir("Opción no válida. Intenta de nuevo.")

    imprimir(f"Seleccionaste la opción {opcion}.")

#Se crea el menú desplegable de las mediciones de longitud.

Si opción es igual a 1:

    imprimir("Ingresa la operación a realizar: \n(1)Centimetros a metros. \n(2)Centimetros a kilometros. \n(3)Metros a centímetros.\n(4)Metros a kilometros. \n(5)kilometros a centimetros. \n(6)kilometros a metros..")

    Repetir hasta que se cumpla:

        medReal = input("Ingresa la operacion a realizar: ")

        Si medReal.Comprobar valor positivo y verdadero and integer(medReal) in range(1, 7):

            medReal = integer(medReal)

            Cerrar cuando se cumple el ciclo

        en caso (contrario) de:

            imprimir("Opción no válida. Intenta de nuevo.")

    Si medReal es igual a 1:

        Repetir hasta que se cumpla:

            cmam = input("Ingresa la longitud en centímetros (mayor que cero): ")

            Completar instrucciones :

                cmam = Ingreso de decimales (cmam)

                Si cmam > 0:

                    Cerrar cuando se cumple el ciclo

                en caso (contrario) de:

                    imprimir("Longitud no válida. Intenta de nuevo.")

            excepción que no cumple try Valor inapropiado:

                imprimir("Longitud no válida. Intenta de nuevo.")

        rmetros = cmam / 100.0

        imprimir(f"{cmam} centímetros equivale a {rmetros} metros.")

    elsi medReal es igual a 2:

        Repetir hasta que se cumpla:

            cmak = input("Ingresa la longitud en centímetros (mayor que cero): ")

            Completar instrucciones :

                cmak = Ingreso de decimales (cmak)

                Si cmak > 0:

                    Cerrar cuando se cumple el ciclo

                en caso (contrario) de:

                    imprimir("Longitud no válida. Intenta de nuevo.")

            excepción que no cumple try Valor inapropiado:

                imprimir("Longitud no válida. Intenta de nuevo.")

        rkilometros = cmak / 100000.0

        imprimir(f"{cmak} centímetros equivale a {rkilometros} kilometros.")

##xd continuacion de 2 o sea 3.

    Si medReal es igual a 3:

        Repetir hasta que se cumpla:

            macm = input("Ingresa la longitud en metros (mayor que cero): ")

            Completar instrucciones :

                macm = Ingreso de decimales (macm)

                Si macm > 0:

                    Cerrar cuando se cumple el ciclo

                en caso (contrario) de:

                    imprimir("Longitud no válida. Intenta de nuevo.")

            excepción que no cumple try Valor inapropiado:

                imprimir("Longitud no válida. Intenta de nuevo.")

        rcm = macm \* 100.0

        imprimir(f"{macm} metros equivale a {rcm}  centímetros.")

    elsi medReal es igual a 4:

        Repetir hasta que se cumpla:

            makm = input("Ingresa la longitud en metros (mayor que cero): ")

            Completar instrucciones :

                makm = Ingreso de decimales (makm)

                Si makm > 0:

                    Cerrar cuando se cumple el ciclo

                en caso (contrario) de:

                    imprimir("Longitud no válida. Intenta de nuevo.")

            excepción que no cumple try Valor inapropiado:

                imprimir("Longitud no válida. Intenta de nuevo.")

        rkm = makm / 1000.0

        imprimir(f"{makm} metros equivale a {rkm} kilometros.")

##4 en adelante

    Si medReal es igual a 5:

        Repetir hasta que se cumpla:

            kacm = input("Ingresa la longitud en kilometros (mayor que cero): ")

            Completar instrucciones :

                kacm = Ingreso de decimales (kacm)

                Si kacm > 0:

                    Cerrar cuando se cumple el ciclo

                en caso (contrario) de:

                    imprimir("Longitud no válida. Intenta de nuevo.")

            excepción que no cumple try Valor inapropiado:

                imprimir("Longitud no válida. Intenta de nuevo.")

        rcen = kacm \* 100000.0

        imprimir(f"{kacm} kilometros equivale a {rcen}  centímetros.")

    elsi medReal es igual a 6:

        Repetir hasta que se cumpla:

            kmam = input("Ingresa la longitud en kilometros (mayor que cero): ")

            Completar instrucciones :

                kmam = Ingreso de decimales (kmam)

                Si kmam > 0:

                    Cerrar cuando se cumple el ciclo

                en caso (contrario) de:

                    imprimir("Longitud no válida. Intenta de nuevo.")

            excepción que no cumple try Valor inapropiado:

                imprimir("Longitud no válida. Intenta de nuevo.")

        rme = kmam \* 1000.0

        imprimir(f"{kmam} kilometros equivale a {rme} metros.")

Si opcion es igual a 2:

    imprimir("Ingresa la operacion a realizar: \n(1)onzas a libras. \n(2)onzas a kilogramos. \n(3)libras a onzas.\n(4)libras a kilogramos. \n(5)kilogramos a onzas. \n(6)kilogramos a libras..")

    Repetir hasta que se cumpla:

        medpeso = input("Ingresa la operacion a realizar: ")

        Si medpeso.Comprobar valor positivo y verdadero and integer(medpeso) in range(1, 7):

            medpeso = integer(medpeso)

            Cerrar cuando se cumple el ciclo

        en caso (contrario) de:

            imprimir("Opción no válida. Intenta de nuevo.")

#Peso

    Si medpeso es igual a 1:

        Repetir hasta que se cumpla:

            ozaLi = input("Ingresa el peso en onzas (mayor que cero): ")

            Completar instrucciones :

                ozaLi = Ingreso de decimales (ozaLi)

                Si ozaLi > 0:

                    Cerrar cuando se cumple el ciclo

                en caso (contrario) de:

                    imprimir("peso no válida. Intenta de nuevo.")

            excepción que no cumple try Valor inapropiado:

                imprimir("peso no válida. Intenta de nuevo.")

        rlibras = ozaLi / 16.0

        imprimir(f"{ozaLi} onzas equivale a {rlibras} libras.")

    elsi medpeso es igual a 2:

        Repetir hasta que se cumpla:

            cmak = input("Ingresa el peso en onzas (mayor que cero): ")

            Completar instrucciones :

                cmak = Ingreso de decimales (cmak)

                Si cmak > 0:

                    Cerrar cuando se cumple el ciclo

                en caso (contrario) de:

                    imprimir("peso no válida. Intenta de nuevo.")

            excepción que no cumple try Valor inapropiado:

                imprimir("peso no válida. Intenta de nuevo.")

        rkilogramos = cmak / 35.274

        imprimir(f"{cmak} onzas equivale a {rkilogramos} kilogramos.")

    Si medpeso es igual a 3:

        Repetir hasta que se cumpla:

            lbaoz = input("Ingresa el peso en libras (mayor que cero): ")

            Completar instrucciones :

                lbaoz = Ingreso de decimales (lbaoz)

                Si lbaoz > 0:

                    Cerrar cuando se cumple el ciclo

                en caso (contrario) de:

                    imprimir("Peso no válida. Intenta de nuevo.")

            excepción que no cumple try Valor inapropiado:

                imprimir("Peso no válida. Intenta de nuevo.")

        rcm = lbaoz \* 16.0

        imprimir(f"{lbaoz} libras equivale a {rcm}  onzas.")

    elsi medpeso es igual a 4:

        Repetir hasta que se cumpla:

            lbakm = input("Ingresa el peso en libras (mayor que cero): ")

            Completar instrucciones :

                lbakm = Ingreso de decimales (lbakm)

                Si lbakm > 0:

                    Cerrar cuando se cumple el ciclo

                en caso (contrario) de:

                    imprimir("Peso no válida. Intenta de nuevo.")

            excepción que no cumple try Valor inapropiado:

                imprimir("Peso no válida. Intenta de nuevo.")

        rkm = lbakm / 2.205

        imprimir(f"{lbakm} libras equivale a {rkm} kilogramos.")

    Si medpeso es igual a 5:

        Repetir hasta que se cumpla:

            kaoz = input("Ingresa peso en kilogramos (mayor que cero): ")

            Completar instrucciones :

                kaoz = Ingreso de decimales (kaoz)

                Si kaoz > 0:

                    Cerrar cuando se cumple el ciclo

                en caso (contrario) de:

                    imprimir("Peso no válida. Intenta de nuevo.")

            excepción que no cumple try Valor inapropiado:

                imprimir("Peso no válida. Intenta de nuevo.")

        roz = kaoz \* 35.274

        imprimir(f"{kaoz} kilogramos equivale a {roz}  onzas.")

    elsi medpeso es igual a 6:

        Repetir hasta que se cumpla:

            kalb = input("Ingresa peso en kilogramos (mayor que cero): ")

            Completar instrucciones :

                kalb = Ingreso de decimales (kalb)

                Si kalb > 0:

                    Cerrar cuando se cumple el ciclo

                en caso (contrario) de:

                    imprimir("Peso no válida. Intenta de nuevo.")

            excepción que no cumple try Valor inapropiado:

                imprimir("Peso no válida. Intenta de nuevo.")

        rlbk = kalb \* 2.205

        imprimir(f"{kalb} kilogramos equivale a {rlbk} libras.")

#Bytes

Si opcion es igual a 3:

    imprimir("Ingresa la operacion a realizar: \n(1)Bytes a Kilobytes. \n(2)Bytes a Megabytes. \n(3)Kilobytes a Bytes.\n(4)Kilobytes a Megabytes. \n(5)Megabytes a Bytes. \n(6)Megabytes a Kilobytes..")

    Repetir hasta que se cumpla:

        medby = input("Ingresa la operacion a realizar: ")

        Si medby.Comprobar valor positivo y verdadero and integer(medby) in range(1, 7):

            medby = integer(medby)

            Cerrar cuando se cumple el ciclo

        en caso (contrario) de:

            imprimir("Opción no válida. Intenta de nuevo.")

    Si medby es igual a 1:

        Repetir hasta que se cumpla:

            byakilo = input("Ingresa los Bytes (mayor que cero): ")

            Completar instrucciones :

                byakilo = Ingreso de decimales (byakilo)

                Si byakilo > 0:

                    Cerrar cuando se cumple el ciclo

                en caso (contrario) de:

                    imprimir("Cantidad no válida. Intenta de nuevo.")

            excepción que no cumple try Valor inapropiado:

                imprimir("Cantidad no válida. Intenta de nuevo.")

        rbKilobytes = byakilo / 1000.0

        imprimir(f"{byakilo} Bytes equivale a {rbKilobytes} Kilobytes.")

    elsi medby es igual a 2:

        Repetir hasta que se cumpla:

            bamg = input("Ingresa el Cantidad en Bytes (mayor que cero): ")

            Completar instrucciones :

                bamg = Ingreso de decimales (bamg)

                Si bamg > 0:

                    Cerrar cuando se cumple el ciclo

                en caso (contrario) de:

                    imprimir("Cantidad no válida. Intenta de nuevo.")

            excepción que no cumple try Valor inapropiado:

                imprimir("Cantidad no válida. Intenta de nuevo.")

        rbMegabytes = bamg / (1\*10)\*\*6

        imprimir(f"{bamg} Bytes equivale a {rbMegabytes} Megabytes.")

    Si medby es igual a 3:

        Repetir hasta que se cumpla:

            kiaby = input("Ingresa el Cantidad en Kilobytes (mayor que cero): ")

            Completar instrucciones :

                kiaby = Ingreso de decimales (kiaby)

                Si kiaby > 0:

                    Cerrar cuando se cumple el ciclo

                en caso (contrario) de:

                    imprimir("Cantidad no válida. Intenta de nuevo.")

            excepción que no cumple try Valor inapropiado:

                imprimir("Cantidad no válida. Intenta de nuevo.")

        rkby = kiaby \* 1000.0

        imprimir(f"{kiaby} Kilobytes equivale a {rkby}  Bytes.")

    elsi medby es igual a 4:

        Repetir hasta que se cumpla:

            kiame = input("Ingresa el Cantidad en Kilobytes (mayor que cero): ")

            Completar instrucciones :

                kiame = Ingreso de decimales (kiame)

                Si kiame > 0:

                    Cerrar cuando se cumple el ciclo

                en caso (contrario) de:

                    imprimir("Longitud no válida. Intenta de nuevo.")

            excepción que no cumple try Valor inapropiado:

                imprimir("Longitud no válida. Intenta de nuevo.")

        rkme = kiame / 1000.0

        imprimir(f"{kiame} Kilobytes equivale a {rkme} Megabytes.")

    Si medby es igual a 5:

        Repetir hasta que se cumpla:

            meaby = input("Ingresa los Megabytes (mayor que cero): ")

            Completar instrucciones :

                meaby = Ingreso de decimales (meaby)

                Si meaby > 0:

                    Cerrar cuando se cumple el ciclo

                en caso (contrario) de:

                    imprimir("Cantidad válida. Intenta de nuevo.")

            excepción que no cumple try Valor inapropiado:

                imprimir("Cantidad no válida. Intenta de nuevo.")

        rmby = meaby \* (1\*10)\*\*6

        imprimir(f"{meaby} Megabytes equivale a {rmby}  Bytes.")

    elsi medby es igual a 6:

        Repetir hasta que se cumpla:

            Meaki = input("Ingresa cantidad en Megabytes (mayor que cero):")

            Completar instrucciones :

                Meaki = Ingreso de decimales (Meaki)

                Si Meaki > 0:

                    Cerrar cuando se cumple el ciclo

                en caso (contrario) de:

                    imprimir("Cantidad no válida. Intenta de nuevo.")

            excepción que no cumple try Valor inapropiado:

                imprimir("Cantidad no válida. Intenta de nuevo.")

        rmki = Meaki \* 1000.0

        imprimir(f"{Meaki} Megabytes equivale a {rmki} Kilobytes.")

si opcionigual4:

    imprimir("Ingresa la operacion a realizar: \n(1)Decimales a binarios. \n(2)Decimales a octal. \n(3)Binario a decimales.\n(4)Binarios a octal. \n(5)octal a decimal. \n(6)Octal a binario.")

    mientras se cumple la condición:

        n = input("Ingresa la operacion a realizar: ")

        si n.isdigit() and int(n) in range(1, 7):

            n = int(n)

            break

        else:

            imprimir("Opción no válida. Intenta de nuevo.")

    si nigual1 :

        def Decimal2binary(numero):

            Binario = ""

            mientras cumpla el ciclo numero // 2 != 0:

                Binario = str (numero%2) + Binario

                numero = numero // 2

            return str(numero) + Binario

        valor\_prueba = int(input("Ingrese el número que desea convertir: "))

        valor\_binario = Decimal2binary(valor\_prueba)

        imprimir ("Su numero {} en binario es {}".ciclomat(valor\_prueba, valor\_binario))

    si nigual2:

        decimal = int(input("Ingresa un número decimal: "))

        octal\_digits = []

        mientras cumpla el ciclo decimal > 0:

            remainder = decimal % 8

            octal\_digits.append(str(remainder))

            decimal = decimal // 8

        octal\_digits.reverse()

        octal = ''.join(octal\_digits)

        imprimir("El número octal correspondiente es:", octal)

    si nigual3:

        numero\_binario = input("Ingrese un número binario: ")

        numero\_decimal = 0

        ciclo i in range(len(numero\_binario)):

            digito = int(numero\_binario[i])

            valor\_decimal = digito \* 2\*\*(len(numero\_binario)-i-1)

            numero\_decimal += valor\_decimal

        imprimir("El número decimal equivalente es:", numero\_decimal)

    si nigual4:

        numero\_binario = input("Ingrese un número binario: ")

        numero\_decimal = 0

        ciclo i in range(len(numero\_binario)):

            digito = int(numero\_binario[i])

            valor\_decimal = digito \* 2\*\*(len(numero\_binario)-i-1)

            numero\_decimal += valor\_decimal

        numero\_octal = oct(numero\_decimal)

        imprimir("El número octal equivalente es:", numero\_octal)

    si nigual5:

        oc = ""

        mientras se cumple la condición:

            oc = input("Ingrese un número octal positivo sin ceros: ")

            si "0" in oc:

                imprimir("Error: el número octal no puede contener ceros.")

            elsi int(oc) < 0:

                imprimir("Error: el número octal debe ser positivo.")

            else:

                break

        nd = 0

        ciclo k in range(len(oc)):

            digito = int(oc[k])

            v\_d = digito \* 8\*\*(len(oc)-k-1)

            nd += v\_d

        imprimir("El número decimal equivalente es:", nd)

    si nigual6:

        def octal\_a\_binario(numoc):

            nudeci = 0

            ciclo i in range(len(numoc)):

                digito = int(numoc[i])

                Vadeci = digito \* 8\*\*(len(numoc)-i-1)

                nudeci += Vadeci

            Nubi = bin(nudeci)[2:]

            return Nubi

        numoc = ""

        mientras cumpla el ciclo numoc igual "" or numoc igual "0":

            numoc = input("Ingrese un número octal distinto de cero: ")

        Nubi = octal\_a\_binario(numoc)

        imprimir("El número binario equivalente es:", Nubi)

        mientras se cumple la condición:

            numoc = input("Ingrese otro número octal distinto de cero (o escriba 'salir' para terminar): ")

            si numoc.lower() igual "salir":

                break

            elsi numoc igual "0":

                imprimir("Error: el número octal no puede ser cero.")

            else:

                Nubi = octal\_a\_binario(numoc)

                imprimir("El número binario equivalente es:", Nubi)

imprimir("Gracias por tu prefencia hasta la proxima  😎😎😎")