ufafuyvpm

January 23, 2025

[ ]:
```
#functions
#if "main"=main:
  #__init__()
```

[ ]:
```
#program to ask user for a price and return the price after discount
def price(price, vat=13):
  return price+(vat/100)*price,price

x,y=price(1000)
print(x,y)
```

```
1130.0 1000
```

[ ]:
```
#args and kwargs
def sample_func(*args,**kwargs):#args takes tuple vakues
  print(args) #kwargs takes dictionary values
  print(kwargs)

sample_func(1,2,3,4,x=2,y=3)
```

```
(1, 2, 3, 4)
{'x': 2, 'y': 3}
```

[ ]:
```
#exceptiton handling
def func():
  y=2
  x=input("Enter the value")
  try:
    return x+y
  except TypeError:
    print("Type error occured")
  except ValueError:
    print("Value out of scope")
  else:
    print("No errors encountered")
  finally:
     print("This is always executed")
```

```
func()
```

```
Enter the valuea
Type error occured
This is always executed
```

[ ]:
```
def func(*args,**raman):
  z= raman.get('dis',0)
  if args[0]>args[1]:
    return (args[0]-args[1])/args[0]*100,(z/100)*args[1]
  else:
    return (args[1]-args[0])/args[0]*100,(z/100)*args[1]

x=int(input("Enter cp"))
y=int(input("Enter sp"))
a,b=func(x,y,dis=2)
print(a,b)
```

```
Enter cp100
Enter sp120
20.0 2.4
```

[ ]:
```
#from math import * #using this we dont need to call the math module evey time
```

[ ]:
```
'''print(ceil(2.4))
print(floor(2.4))
print(sqrt(9))'''
```

```
3
2
3.0
```

[ ]:
```
#recursion function
def funcn(n):
  if n==0 or n==1:
    return 1
  else:
    return n*funcn(n-1)

funcn(5)
```

[ ]:
```
120
```

[ ]:
```
def suum(n):
  if n==1:
    return 1
  else:
```

```
    return n+suum(n-1)

suum(10)
```

[ ]: 55

[ ]:
```
def lst(n):
  even=[]
  odd=[]
  for i in range(1,n+1):
    if i%2==0:
      even.append(i)
    else:
      odd.append(i)
  return even,odd
a,b=lst(10)
print(a,b)
```

```
[2, 4, 6, 8, 10] [1, 3, 5, 7, 9]
```

Modules

[ ]: