```
{
 "cells": [
  {
   "cell_type": "markdown",
   "id": "a7a90026",
   "metadata": {},
   "source": [
    "Q11)Write a python program to find the factorial of a number.\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 1,
   "id": "34bc0aab",
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "enter a number:25\n",
      "1525"
     ]
    }
   ],
   "source": [
    "n=int(input(\"enter a number:\"))\n",
    "for i in range(1,n+1): \n",
    "    if not n%i:\n",
    "        print(i,end=\"\")\n"
   ]
  },
  {
   "cell_type": "markdown",
   "id": "adb8c9a2",
   "metadata": {},
   "source": [
    "Q12)Write a python program to find whether a number is prime or
composite"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 6,
   "id": "fb602363",
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "enter a number:467\n",
      "467 is not a prime number\n"
     ]
    }
   ],
   "source": [
    "num=int(input(\"enter a number:\"))\n",
```

```
    "if num>1:\n",
    "    for i in range(2,num):\n",
    "        print(num,\"is not a prime number\")\n",
    "        break\n",
    "    else:\n",
    "         print(num,\"is a prime number\")\n",
    "elif num==0 or 1:\n",
    "    print(num,\"is neither prime nor a composite number\")\n",
    "else:\n",
    "    print(num,\"is a composite number not a prime number\")\n",
    "\n",
    "    "
   ]
  },
  {
   "cell_type": "markdown",
   "id": "7652a9cd",
   "metadata": {},
   "source": [
    "Q13) Write a python program to check whether a given string is palindrome or not"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 16,
   "id": "d2243dab",
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "yes\n"
     ]
    }
   ],
   "source": [
    "def isPalindrome(c):\n",
    "    return c==c[::-1]\n",
    "c=\"kayak\"\n",
    "ans=isPalindrome(c)\n",
    "\n",
    "if ans:\n",
    "    print(\"yes\")\n",
    "else:\n",
    "    print(\"no\")\n",
    "        "
   ]
  },
  {
   "cell_type": "markdown",
   "id": "0f117f91",
   "metadata": {},
   "source": [
    "Q14)Write a python program to get the third side of right angled triangle from two given sides"
   ]
```

```
  },
  {
   "cell_type": "code",
   "execution_count": 17,
   "id": "87adcfcc",
   "metadata": {},
   "outputs": [],
   "source": [
    "import math"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 19,
   "id": "12db67f5",
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "base:5\n",
      "height:6\n",
      "hypotenuse= 7.810249675906654\n"
     ]
    }
   ],
   "source": [
    "b=float(input(\"base:\"))\n",
    "p=float(input(\"height:\"))\n",
    "h=math.sqrt(b**2 + p**2)\n",
    "print(\"hypotenuse=\",h)\n"
   ]
  },
  {
   "cell_type": "markdown",
   "id": "210adbe4",
   "metadata": {},
   "source": [
    "Q15)Write a python program to print the frequency of each of the
characters present in a given string\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 24,
   "id": "353b5899",
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "count of all characters in MachineLearning is:\n",
      "{'M': 1, 'a': 2, 'c': 1, 'h': 1, 'i': 2, 'n': 3, 'e': 2, 'L': 1,
'r': 1, 'g': 1}\n"
     ]
    }
```

```
   ],
   "source": [
    "all_freq={}\n",
    "test_str=\"MachineLearning\"\n",
    "for i in test_str:\n",
    "    if i in all_freq:\n",
    "        all_freq[i]+=1\n",
    "    else:\n",
    "        all_freq[i]=1\n",
    "print(\"count of all characters in MachineLearning is:\\n\"+str
(all_freq))        \n",
    "    \n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "9168eb56",
   "metadata": {},
   "outputs": [],
   "source": []
  }
 ],
 "metadata": {
  "kernelspec": {
   "display_name": "Python 3",
   "language": "python",
   "name": "python3"
  },
  "language_info": {
   "codemirror_mode": {
    "name": "ipython",
    "version": 3
   },
   "file_extension": ".py",
   "mimetype": "text/x-python",
   "name": "python",
   "nbconvert_exporter": "python",
   "pygments_lexer": "ipython3",
   "version": "3.8.8"
  }
 },
 "nbformat": 4,
 "nbformat_minor": 5
}
```