tugaspratikumaminatul

May 25, 2024

[ ]:
```
@@ -0,0 +1,377 @@
{
 "cells": [
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "<h2>Pengertian</h2>\n",
    "<ol>\n",
    "    <li>Fungsi pada python adalah kumpulan perintah atau baris kode yang dikelompokkan menjadi satu kesatuan untuk kemudian bisa dipanggil atau digunakan berkali-kali.</li>\n",
    "    <li>Sebuah fungsi dapat menerima parameter, dapat mengembalikan suatu nilai, dan dapat dipanggil berkali-kali secara independen.</li>\n",
    "</ol>\n",
    "<hr>\n",
    "<h2>Keuntungan</h2>\n",
    "<ol>\n",
    "    <li>Membagi kode program menjadi bagian-bagian kecil dengan tugasnya masing-masing.</li>\n",
    "    <li>Membuat kode program menjadi lebih "reusable" dan lebih terstruktur.</li>\n",
    "</ol>\n",
    "<hr>\n",
    "<h2>Sintaks</h2>\n",
    "\n",
    "```python\n",
    "def function_name():\n",
    "    Logic Program\n",
    "```\n",
    "Cara memanggil fungsi dengan menuliskan function_name diikuti kurung buka tutup () beserta parameter jika ada\n",
    "\n",
    "```python\n",
    "    function_name()\n",
    "```\n",
    "<hr>\n",
```

```
   "Buatlah fungsi dengan nama \"helloPython\" yang berfungsi untuk mencetak␣
↪\"Welcome in Python Language\" "
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 1,
  "metadata": {},
  "outputs": [
   {
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "Welcome in Python Language\n"
    ]
   }
  ],
  "source": [
   "def hello_python():\n",
   "    print(\"Welcome in Python Language\")\n",
   "\n",
   "hello_python()"
  ]
 },
 {
  "cell_type": "markdown",
  "metadata": {},
  "source": [
   "<h1>Fungsi dengan parameter</h1>\n",
   "Sebuah fungsi dapat menerima parameter ataupun argumen yang merupakan␣
↪suatu nilai/variabel yang dilemparkan ke dalam fungsi untuk diproses lebih␣
↪lanjut.\n",
   "\n",
   "<h2>Sintaks</h2>\n",
   "\n",
   "```python\n",
   "def function_name(param):\n",
   "    Logic Program\n",
   "```\n",
   "Cara memanggil fungsi dengan menuliskan function_name diikuti kurung buka␣
↪tutup () beserta parameter\n",
   "\n",
   "```python\n",
   "    function_name(param)\n",
   "```\n",
   "<hr>"
  ]
```

```
 },
 {
  "cell_type": "markdown",
  "metadata": {},
  "source": [
   "Buatlah fungsi dengan nama \"fullName\" yang mempunyai parameter␣
↪\"firstname\" dan \"lastname\" digunakan untuk mencetak \"firstname\" dan␣
↪\"lastname\""
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 5,
  "metadata": {},
  "outputs": [
   {
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "His First Name Is : Rangga\n",
     "His Last Name Is : Hishbu Shafar\n"
    ]
   }
  ],
  "source": [
   "def fullname(x, y):\n",
   "    print(f\"His First Name Is : {x}\")\n",
   "    print(f\"His Last Name Is : {y}\")\n",
   "\n",
   "fullname(\"Rangga\", \"Hishbu Shafar\")"
  ]
 },
 {
  "cell_type": "markdown",
  "metadata": {},
  "source": [
   "Parameter fungsi diperbolehkan menggunakan lebih dari 1, dimana paramater␣
↪tersebut ada yang wajib harus diisi dan ada yang tidak harus diisi\n",
   "\n",
   "```python\n",
   "def function_name(param_1, param_2, param_3,....):\n",
   "    Logic Program\n",
   "```\n",
   "\n",
   "Cara memanggil fungsi dengan menuliskan function_name diikuti kurung buka␣
↪tutup () beserta parameter\n",
   "\n",
```

```
   "```python\n",
   "    function_name(param_1, param_2, param_3)\n",
   "```\n",
   "<hr>\n",
   "Buatlah fungsi dengan nama maxValue yang mempunyai parameter␣
↪\"val_1\",\"val_2\" dan \"val_3\" yang bertujuan untuk mencari nilai␣
↪terbesar dari 3 nilai tersebut"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 12,
  "metadata": {},
  "outputs": [
   {
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "12\n"
    ]
   }
  ],
  "source": [
   "def max_value(val_1, val_2, val_3):\n",
   "    print(max(val_1, val_2, val_3))\n",
   "\n",
   "max_value(5,12,9)"
  ]
 },
 {
  "cell_type": "markdown",
  "metadata": {},
  "source": [
   "Parameter opsional digunakan pada fungsi dengan cara memberikan nilai␣
↪default, artinya nilai parameter sudah diberikan terlebih dahulu tanpa␣
↪dipanggil\n",
   "\n",
   "```python\n",
   "def function_name(param_1, param_2, param_3 = 'Nilai'):\n",
   "    Logic Program\n",
   "```"
  ]
 },
 {
  "cell_type": "markdown",
  "metadata": {},
  "source": [
```

```
   "Buatlah fungsi \"countCircleArea\" dengan 2 parameter yaitu \"phi\" dan␣
↪\"diameter\" dimana parameter phi mempunyai nilai default 3.14"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 18,
  "metadata": {},
  "outputs": [
   {
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "Luas lingkaran :  153.86\n"
    ]
   }
  ],
  "source": [
   "def lingkaran(diameter, phi = 3.14):\n",
   "    luas = phi * diameter * diameter\n",
   "    print(f\"Luas lingkaran : \", luas)\n",
   "\n",
   "lingkaran(7)"
  ]
 },
 {
  "cell_type": "markdown",
  "metadata": {},
  "source": [
   "Fungsi dengan 2 Parameter opsional juga dapat dilakukan pada Python. \n",
   "\n",
   "```python\n",
   "def info(suhu, daerah='Sukabumi', satuan = 'Celcius'):\n",
   "    print(f\"Suhu sekarang di {daerah} : {suhu} {satuan} \")\n",
   "```\n",
   "Adapun Cara memanggilnya\n",
   "\n",
   "```python\n",
   "    info(30)\n",
   "```"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 17,
  "metadata": {},
  "outputs": [
```

```
   {
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "Suhu sekarang di Sukabumi : 30 Celcius \n"
    ]
   }
  ],
  "source": [
   "#Cobalah fungsi tersebut pada area kode disini\n",
   "def info(suhu, daerah='Sukabumi', satuan = 'Celcius'):\n",
   "    print(f\"Suhu sekarang di {daerah} : {suhu} {satuan} \")\n",
   "\n",
   "info(30)"
  ]
 },
 {
  "cell_type": "markdown",
  "metadata": {},
  "source": [
   "<h1>Fungsi dengan Return Value</h1>\n",
   "\n",
   "Yaitu fungsi dimana akhir dari programnya adalah nilai kembalian atau␣
↪nilai balik. Artinya nilai dalam fungsi dapat ditampung lagi ke variabel␣
↪lain untuk digunakan operasi lebih lanjut.\n",
   "\n",
   "Buatlah fungsi dengan return value yang digunakan untuk mengecek sebuah␣
↪bilangan termasuk bilangan negatif, bilangan neteral(0) dan bilangan positif"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 24,
  "metadata": {},
  "outputs": [
   {
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "Ini Negatif\n",
     "Ini Netral\n",
     "Ini Positif\n",
     "Ini Negatif\n"
    ]
   }
  ],
  "source": [
```

```
   "def cek_angka(angka):\n",
   "    if angka < 0:\n",
   "        return \"Ini Negatif\"\n",
   "\n",
   "    elif angka == 0:\n",
   "        return \"Ini Netral\"\n",
   "\n",
   "    else:\n",
   "        return \"Ini Positif\"\n",
   "\n",
   "print(cek_angka(-6))\n",
   "print(cek_angka(0))\n",
   "print(cek_angka(10))\n",
   "print(cek_angka(-100))"
  ]
 },
 {
  "cell_type": "markdown",
  "metadata": {},
  "source": [
   "<h3>Soal Latihan</h3>\n",
   "<ol>\n",
   "    <li>Buatlah fungsi untuk menjumlahkan total nilai dari list</li>\n",
   "    <li>Buatlah fungsi untuk mencari nilai terbesar dari sekumpulan list </
↪li>\n",
   "    <li>Buatlah fungsi untuk menjumlahkan 2 buah list </li>\n",
   "</ol>\n"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 41,
  "metadata": {},
  "outputs": [
   {
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "Total nilai adalah =  18\n"
    ]
   }
  ],
  "source": [
   "#Soal 1\n",
   "def nilai(angka):\n",
   "    total = sum(angka)\n",
   "    return total\n",
```

```
   "\n",
   "list = [1,2,4,5,6]\n",
   "total_jum = nilai(list)\n",
   "print(f\"Total nilai adalah = \",total_jum)"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 40,
  "metadata": {},
  "outputs": [
   {
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "Angka terbesar adalah =  90\n"
    ]
   }
  ],
  "source": [
   "#Soal 2\n",
   "def nilai_terbesar(number):\n",
   "    besar = max(number)\n",
   "    return besar\n",
   "\n",
   "numbers = [5,12,90,21]\n",
   "terbesar = nilai_terbesar(numbers)\n",
   "print(f\"Angka terbesar adalah = \", terbesar)"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 12,
  "metadata": {},
  "outputs": [
   {
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "Hasil Penjumlahan: [5, 5, 5, 5]\n"
    ]
   }
  ],
  "source": [
   "#Soal 3\n",
   "list1 = [1,2,3,4]\n",
   "list2 = [4,3,2,1]\n",
```

```
    "\n",
    "def jum_list(list_1, list_2):\n",
    "    hasil = []\n",
    "    for a, b in zip(list_1, list_2):\n",
    "        hasil.append(a + b)\n",
    "\n",
    "    return hasil \n",
    "\n",
    "hasil_penjumlahan = jum_list(list1, list2)\n",
    "print(\"Hasil Penjumlahan:\", hasil_penjumlahan)"
   ]
  }
 ],
 "metadata": {
  "kernelspec": {
   "display_name": "Python 3 (ipykernel)",
   "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": "3.11.7"
  }
 },
 "nbformat": 4,
 "nbformat_minor": 4
}
```