Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	**71230982**
Nama Lengkap	**Rainie Fanita Chrisabel Hadisantoso**
Minggu ke / Materi	**05 / Struktur Kontrol Perulangan**

SAYA MENYATAKAN BAHWA LAPORAN PRAKTIKUM INI SAYA BUAT DENGAN USAHA SENDIRI TANPA MENGGUNAKAN BANTUAN ORANG LAIN. SEMUA MATERI YANG SAYA AMBIL DARI SUMBER LAIN SUDAH SAYA CANTUMKAN SUMBERNYA DAN TELAH SAYA TULIS ULANG DENGAN BAHASA SAYA SENDIRI.

SAYA SANGGUP MENERIMA SANKSI JIKA MELAKUKAN KEGIATAN PLAGIASI, TERMASUK SANKSI TIDAK LULUS MATA KULIAH INI.

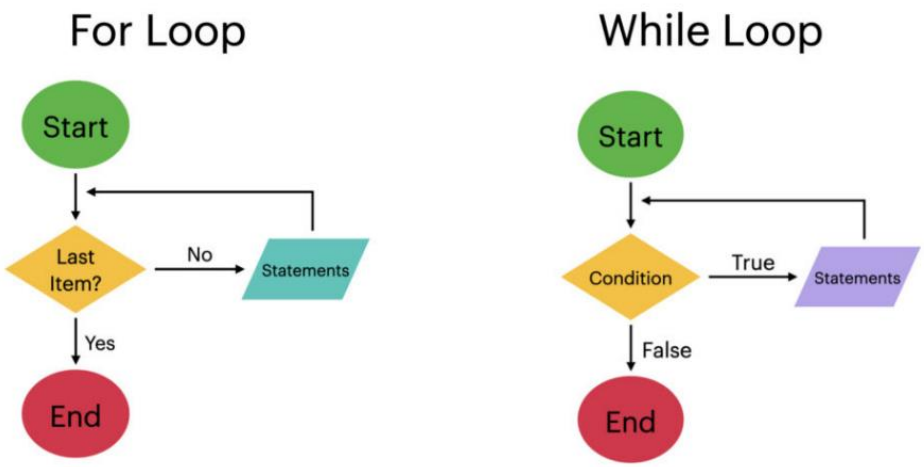
PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS KRISTEN DUTA WACANA
YOGYAKARTA
2024

MATERI: Struktur Kontrol Perulangan

Pelajaran praktikum ini bertujuan untuk mempelajari bentuk perulangan for dan while dan menggunakannya, menggunakan break dan continue dalam perulangan, mengubah perulangan bentuk for menjadi while dan menerapkan semuanya untuk menyelesaikan masalah.

Definisi Perulangan

Perulangan merupakan salah satu struktur kontrol yang dapat digunakan untuk mengatur jalur kerja program. Seperti sebutannya, perulangan digunakan untuk melakukan suatu set perintah yang perlu dilakukan berulang-ulang. Biasanya, perulangan digunakan ketika program perlu melakukan urutan perintah yang sama beberapa kali. Selain itu perulangan juga digunakan untuk mengolah data-data dalam sebuah struktur data atau array seperti List, Tuple, Queue, Stack, dan lainya.



Gambar 1. Struktur Flowchart For dan While

Dalam Python, terdapat beberapa cara menggunakan perulangan seperti menggunakan for, while maupun rekursif. Beberapa bentuk perulangan dapat digunakan bersama-sama dalam sebuah program, biasanya disebut dengan nested loop.

No	Tipe Loop	Penjelasan
1.	While loop	Perulangan dilakukan selama keadaan masih TRUE, akan dilakukan pengecekan kondisi terlebih dahulu sebelum blok kode dieksekusi.
2.	For loop	Eksekusi terhadap blok kode dilakukan berulang kali sesuai dengan variabel yang mengatur perulangan.
3.	Nested loop	Kita bisa mengkombinasikan perulangan, dimana ada perulangan di dalam perulangan.

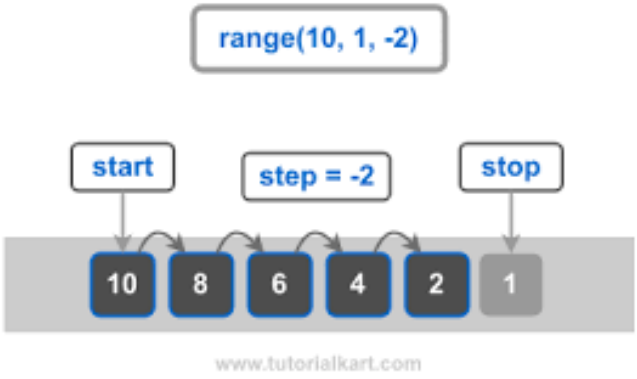
Gambar 2. Tabel Perbedaan Bentuk Perulangan

Dalam pertemuan praktikum ini, kita hanya mempelajari penggunaan perulangan for dan while dan mengaplikasikannya dalam berbagai algoritma untuk memecahkan masalah yang diberikan.

Bentuk Perulangan for

Bentuk perulangan for biasanya digunakan ketika kita mengetahui dengan jelas sampai berapa kali perulangan hendak dilakukan. Selain itu kita dapat menggunakan for ketika ingin melakukan perulangan dalam sebuah rentang batasan data maupun nilai. For juga dapat digunakan untuk mengakses data-data dalam sebuah array.

Kita dapat membatasi perulangan for dalam rentang tertentu menggunakan fungsi range (start, stop, step). Start adalah kapan perulangan mulai, Stop untuk kapan perulangan berhenti, dan step untuk berapa lompatan yang dilakukan perulangan setiap kalinya.



Gambar 3. Start, Stop, Step

Dalam Python, step memiliki nilai default yaitu bertambah 1, sehingga penggunaan fungsi range dapat hanya menggunakan start dan stop. Start juga memiliki nilai default 0, sehingga fungsi range dapat hanya menggunakan stop.

```
1   for bilangan in range (10):
2       print(bilangan, end=" ")
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\asus\belajar-git> python -u "c:\Users\asus\belajar-git\belajar python
 2\alpro5.2.py"
0 1 2 3 4 5 6 7 8 9
```

Gambar 4. For Stop

Untuk kasus tertentu, diperlukan ketiga kondisi fungsi range untuk menjalankan perulangan for. Seperti ketika fungsi membutuhkan nilai step maupun start yang berbeda dari nilai defaultnya.

```
4   for bilangan in range (1,10,2):
5       print(bilangan, end=" ")
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\asus\belajar-git> python -u "c:\Users\asus\belajar-git\belajar python
 2\alpro5.2.py"
1 3 5 7 9
```

Gambar 5.For Start, Stop, Step

Fungsi range juga dapat mengolah data secara mundur, khususnya data berupa angka. Untuk melakukan pengolahan data bilangan secara mundur kita dapat menggunakan nilai step negatif. Dengan begitu nilai start akan terus-menerus dikurangi dengan nilai step yang diberikan hingga mencapai batas stop yang ada. Biasanya dalam hal ini, nilai start lebih besar dari nilai stop.

```
7   for bilangan in range (10,0,-1):
8       print(bilangan, end=" ")
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\asus\belajar-git> python -u "c:\Users\asus\belajar-git\belajar python
 2\alpro5.2.py"
10 9 8 7 6 5 4 3 2 1
```

Gambar 6. For Step Negatif

Berikut contoh program Python untuk mencari bilangan ganjil dalam rentang 0 sampai 100 menggunakan perulangan for.

```
1   print("Ini ganjil for")
2   for bilangan in range (0, 100, 1):
3       if bilangan % 2 != 0:
4           print(bilangan, end=" ")
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\asus\belajar-git> python -u "c:\Users\asus\belajar-git\belajar python
 2\alpro5.2.py"
Ini ganjil for
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57
 59 61 63 65 67 69 71 73 75 77 79 81 83 85 87 89 91 93 95 97 99
```

Gambar 7. Program Ganjil For

Bentuk Perulangan while

Bentuk perulangan while digunakan ketika kita tidak mengetahui kapan perulangan harus berhenti. Karena itu biasanya perulangan while menggunakan batas kondisi-kondisi tertentu untuk menghentikan perulangan. Kondisi dapat menggunakan True dan False, operator perbandingan, dan lain-lain. Berikut struktur perulangan while secara umum.

```
<start>
```

```
while <stop>:
    operation
    operation
    <step (optional)>
```

Berikut contoh program python untuk menampilkan bilangan ganjil dalam rentang 0 sampai 100 menggunakan perulangan while.

```
1   print("Ini ganjil while:")
2   bilangan = 0
3   while bilangan <= 100:
4       if bilangan % 2 != 0:
5           print(bilangan, end=" ")
6       bilangan += 1
7
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\asus\belajar-git> python -u "c:\Users\asus\belajar-git\belajar python
 2\alpro5.1.py"
Ini ganjil while:
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57
 59 61 63 65 67 69 71 73 75 77 79 81 83 85 87 89 91 93 95 97 99
```

Gambar 8. Program Ganjil While

Dalam penggunaan perulangan while kita juga dapat menambahkan lanjutan lain ketika kondisi tidak terpenuhi menggunakan else:

```
<start>
while <stop>:
    operation
    operation
    <step (optional)>
Else:
    Operation
```

Contoh program Python menampilkan bilangan ganjil menggunakan perulangan while-else:

```
1   print("Ini ganjil while:")
2   bilangan = 0
3   while bilangan <= 100:
4       if bilangan % 2 != 0:
5           print(bilangan, end=" ")
6       bilangan += 1
7   else:
8       print("Sudah selesai")
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\asus\belajar-git> python -u "c:\Users\asus\belajar-git\belajar python
 2\alpro5.1.py"
Ini ganjil while:
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57
 59 61 63 65 67 69 71 73 75 77 79 81 83 85 87 89 91 93 95 97 99 Sudah selesai
```

Gambar 9. Program Ganjil While Else

Ketika nilai variable sudah melebihi 100, perulangan while akan berhenti dan program masuk ke dalam else, sehingga ditampilkan “Sudah selesai” di bagian paling akhir output.

Penggunaan Break dan Continue

Jalannya perulangan dapat diatur dengan perintah break dan continue. Secara singkat break digunakan untuk menghentikan jalannya perulangan ketika kondisi-kondisi yang diberikan terpenuhi. Sedangkan continue digunakan untuk melakukan skip atau melewati perintah perulangan ketika kondisi terpenuhi dan kemudian program akan melanjutkan perulangan hingga batas stop. Berikut contoh perbedaan hasil penggunaan break dan stop pada program menampilkan bilangan 0 hingga 10.

1. Output break:

```
1  for bilangan in range (10):
2      if bilangan == 5:
3          break
4      else:
5          print(bilangan, end=" ")
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\asus\belajar-git> python -u "c:\Users\asus\belajar-git\belajar python
 2\alpro5.2.py"
0 1 2 3 4
```

Gambar 10. Contoh Break

2. Output continue:

```
1  for bilangan in range (10):
2      if bilangan == 5:
3          continue
4      else:
5          print(bilangan, end=" ")
```

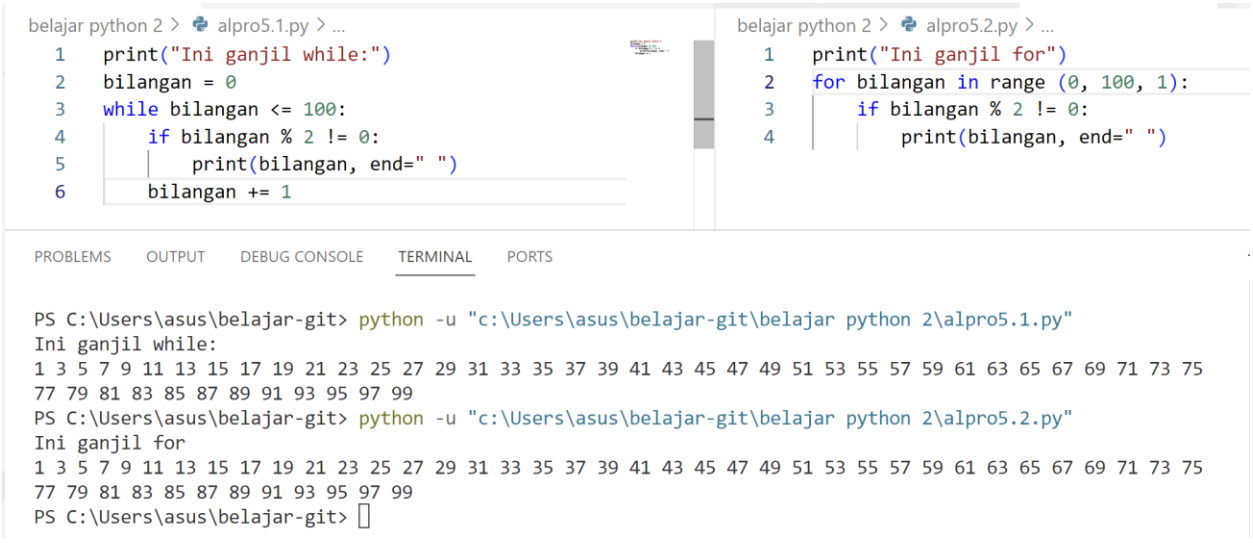
```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\asus\belajar-git> python -u "c:\Users\asus\belajar-git\belajar python
 2\alpro5.2.py"
0 1 2 3 4 6 7 8 9
```

Gambar 11. Contoh Continue

Dapat dilihat hasil dari program break hanya menampilkan bilangan 0 sampai 4, padahal stopnya berada di 10. Hal tersebut terjadi karena ketika bilangan mencapai nilai 5 perulangan dihentikan oleh break, sehingga program tidak menampilkan bilangan 5 sampai 10. Sedangkan dalam program continue, program memberikan output 0, 1, 2, 3, 4, 6, 7, 8, 9, 10, namun tidak menampilkan bilangan 5. Hal tersebut terjadi karena ketika bilangan bernilai 5 perulangan ter-skip karena continue dan lanjut menampilkan bilangan setelahnya.

Konversi dari Bentuk for Menjadi Bentuk while

Kesamaan yang dimiliki oleh perulangan for dan while, kecuali kegunaannya, adalah memiliki sistem start, stop, dan step. Karena itu memungkinkan agar bentuk for diubah ke while dan bentuk while diubah ke for. Namun hal tersebut hanya dapat dilakukan ketika ketiga kondisi start, stop, dan step terpenuhi dalam bentuk perulangan masing-masing. Berikut contoh program Python menampilkan bilangan genap dari 0 sampai 100 dalam bentuk for dan while. Dapat dilihat kedua program dapat menghasilkan output yang sama dengan menggunakan bentuk perulangan yang berbeda.



Gambar 12. Program Bentuk For dan While

LATIHAN 5

SOAL 5.1

Membuat program perkalian dengan cara penjumlahan dengan fungsi perkalian(). Contoh hasil dari beberapa test case:

- 6 x 5 = 5 + 5 + 5 + 5 + 5 + 5 = 30.
- 7 x 10 = 10 + 10 + 10 + 10 + 10 + 10 + 10 = 70.

a. Source Code

```
def perkalian( kali, bilangan ):
  print(f"{kali} x {bilangan} =", end=" ")
  hasil = 0
  for i in range(kali):
    if i < kali-1:
      print(bilangan, end=" + ")
    elif i == kali-1:
      print(bilangan, end=" ")

    hasil += bilangan

  print(f"= {hasil}.")

kali = int(input("Masukkan pengali= "))
bilangan = int(input("Masukkan bilangan= "))

perkalian(kali, bilangan)
```

b. Output

```
Masukkan pengali= 6
Masukkan bilangan= 5
6 x 5 = 5 + 5 + 5 + 5 + 5 + 5 = 30.
```

```
Masukkan pengali= 7
Masukkan bilangan= 10
7 x 10 = 10 + 10 + 10 + 10 + 10 + 10 + 10 = 70.
```

c. Penjelasan

Dalam fungsi perkalian:

- Untuk print pertama digunakan untuk menampilkan bilangan pengali dan dikali.
- Variabel hasil digunakan untuk nanti menghitung pertambahan. Hasil bernilai 0 agar hasil sesuai dengan jumlah pengulangan yang dilakukan

- Perulangan sebanyak bilangan pengali. Dalam pengulangan terdapat percabangan: If untuk menampilkan bilangan dengan koma (,), elif agar perulangan terakhir menampilkan bilangan tanpa titik. Selain itu ada penambahan hasil dengan bilangan dan print menampilkan = hasil.
- Dari keseluruhan fungsi hasil yang akan ditampilkan: angka pengali x angka dikali = angka dikali + angka dikali + ... (diulang sesuai pengali) ...+ angka dikali + angka dikali = hasil.

Di luar fungsi perkalian:

- Di luar fungsi ada variabel kali untuk memasukkan bilangan pengali int dari pengguna dan bilangan untuk bilangan dikali int dari pengguna.
- Terakhir fungsi perkalian dipanggil.

SOAL 5.2

Membuat program menampilkan deret bilangan ganjil dengan batas bawah dan batas atas yang dimasukkan oleh pengguna. Program menggunakan fungsi dengan nama ganjil(). Jika batas bawah > batas bawah maka bilangan ganjil ditampilkan dengan terbalik dan sebaliknya. Beberapa contoh output dari test case:

- bawah = 10, atas = 30. Karena bawah < atas, berarti dari kecil ke besar, maka hasilnya adalah: 11, 13, 15, 17, 19, 21, 23, 25, 27, 29.
- bawah = 97, atas = 82. Karena bawah > atas, berarti dari besar ke kecil, maka hasilnya adalah: 97, 95, 93, 91, 89, 87, 85, 83.

a. Source Code

```
def ganjil(bawah, atas):
  if bawah <= atas:
    for i in range(bawah, atas):
      if i % 2 != 0:
        if i < atas - 1:
          print(i, end=", ")
        elif i == atas - 1:
          print(i, end=".")
  else:
    for i in range(bawah, atas, -1):
      if i % 2 != 0:
        if i > atas + 1:
          print(i, end=", ")
        elif i == atas + 1:
          print(i, end=".")

bawah = int(input("Masukkan bilangan awal= "))
atas = int(input("Masukkan bilangan akhir= "))

ganjil(bawah,atas)
```

b. Output

```
Masukkan bilangan awal= 10
Masukkan bilangan akhir= 30
11, 13, 15, 17, 19, 21, 23, 25, 27, 29.
```

```
Masukkan bilangan awal= 97
Masukkan bilangan akhir= 82
97, 95, 93, 91, 89, 87, 85, 83.
```

c. Penjelasan

Dalam fungsi ganjil

1. Percabangan pertama untuk membedakan range perulangan berdasarkan bilangan bawah dan bilangan atas. Jika bilangan bawah <= dari atas maka perulangan bertambah. Jika bawah >= atas perulangan berkurang.
2. Perulangan bawah <= atas: mencari bilangan ganjil (bilangan % 2 != 0). Ketika bilangan memenuhi kondisi bilangan ganjil maka ada percabangan untuk menampilkan bilangan tersebut. Jika perulangan terjadi berjumlah <= bilangan atas – 1, program akan menampilkan bilangan dengan koma (,), jika sudah sama dengan bilangan atas akan ditampilkan bilangan dengan titik (.) untuk menutup kalimat output.
3. Perulangan bawah >= atas: mencari bilangan ganjil. Kebalikan dengan perulangan pertama, perulangan kedua menggunakan batasan tambahan step -1. Dengan step -1 program akan mengurangi angka terus menerus, sehingga bilangan ganjil yang ditampilkan akan berubah dari besar ke kecil sesuai batas bawah dana atas. Untuk percabangan yang digunakan konsepnya sama dengan percabangan perulangan pertama agar bilangan dipisah dengan koma (,) dan diakhiri dengan titik (.).

Di luar fungsi:

1. Variabel bawah untuk input batas bilangan bawah dan variabel atas untuk input batas bilangan atas.
2. Memanggil fungsi ganjil.

SOAL 5.3

Membuat program untuk menghitung nilai Indeks Prestasi Semester (IPS) menggunakan kontrol percabangan di dalam perulangan. Program membutuhkan input: Jumlah mata kuliah dan Nilai A, B, C, dan D untuk setiap mata kuliah. SKS setiap mata kuliah selalu diasumsikan berbobot 3 SKS dan harga nilai: A=4, B=3, C=2, D=1. Contoh output IPS sebagai berikut:

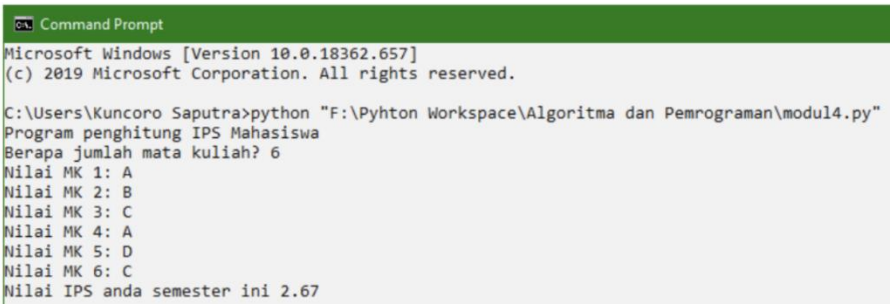


Figure 1. Contoh Output Program IPS

a. Source Code

```
def IPS(matkul, nilai):
  jumlah = 0
  for huruf in nilai:
    if huruf == "A":
      jumlah += 4
    elif huruf == "B":
      jumlah += 3
    elif huruf == "C":
      jumlah += 2
    elif huruf == "D":
      jumlah += 1

  IPS = jumlah / matkul
  print("Nilai IPS anda semester ini =", IPS)

nilai = []
matkul = int(input("Masukkan jumlah mata kuliah = "))
for i in range (matkul):
    i += 1
    huruf = input(f"Nilai MK {i}= ")
    nilai.append(huruf)

IPS(matkul, nilai)
```

b. Output

```
Masukkan jumlah mata kuliah = 4
Nilai MK 1= A
Nilai MK 2= B
Nilai MK 3= C
Nilai MK 4= D
Nilai IPS anda semester ini = 2.5
```

c. Penjelasan

Dalam fungsi IPS:

1. Variabel jumlah dengan nilai 0. Nantinya variabel ini akan digunakan untuk penambahan harga nilai berdasarkan nilai yang di input.
2. Perulangan untuk mengidentifikasi nilai huruf sesuai dengan harga nilanya masing-masing menggunakan percabangan. Nilai-nilai huruf yang di input pengguna disimpan dalam array bernama nilai. Dalam percabangan, variabel jumlah ditambahkan dengan harga nilai sesuai dengan data nilai huruf.
3. Variabel IPS untuk menghitung nilai semester. Ketika semua harga nilai huruf sudah dijumlahkan, hasil total akan dibagi dengan jumlah mata kuliah yang diambil oleh pengguna untuk mendapat rata-rata harga nilai. Karena asumsi semua mata kuliah berbobot 3 SKS, perkalian harga nilai berdasarkan bobot SKS tidak diperhitungkan.
4. Fungsi print untuk menampilkan hasil IPS.

Di luar fungsi:

1. Variabel nilai didefinisikan sebagai array = []
2. Variabel matkul untuk memasukkan jumlah mata kuliah yang diambil oleh pengguna.
3. Perulangan dengan batas jumlah mata kuliah untuk meng-input nilai huruf yang didapat pengguna sesuai dengan jumlah mata kuliah yang diambil. Input nilai huruf kemudian dimasukkan dalam array nilai dengan .perintah append().
4. Dipanggil fungsi IPS.

Link Github:

https://github.com/rainiefch/Praktikum-Algoritma-Pemrograman_71230982.git