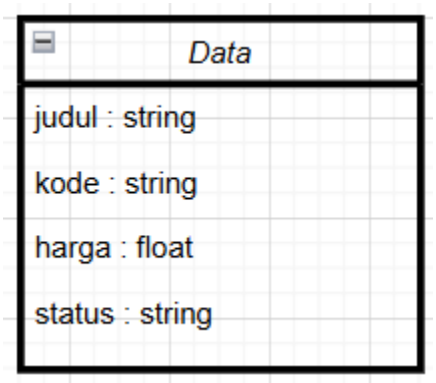
UNGUIDED STACK ARRAY
TIPE E

PT Widhari merupakan sebuah perusahaan besar yang bergerak di bidang penjualan buku. Rendy yang merupakan bos dari PT Widhari merupakan orang yang sangat dermawan dan suka dengan permainan kata. Tiap buku yang kodenya bersifat **Palindrom** akan **secara otomatis mendapatkan potongan harga sebesar 25%**.

Palindrom adalah adalah kata, frasa, dan angka yang akan terdengar sama walaupun dibaca dari belakang. Contohnya : Tenet, Kasur ini rusak, Ada.

Anda, sebagai programmer mendapatkan tugas untuk membuat program yang secara otomatis **memberikan diskon** pada **buku yang kodenya memiliki sifat Palindrom**, serta dapat menyimpan data yang banyak **(Maksimal 5 Data saja)**. Berikut menu dan ketentuan dari program tersebut:



Gambar 1. Diagram Kelas

1) Input Data (20 poin)
   Menu ini digunakan untuk menginputkan data yang diperlukan berupa judul (*string*), kode (*string*), harga (*float*), dan status (*string*). Kemudian, kelima data tersebut akan disimpan ke dalam ***array***. Menu ini **tidak bisa diakses jika data pada array sudah penuh**. Ada juga ketentuan dalam pengisian data-data tersebut, antara lain :
   1. *Input* **judul buku** tidak boleh kosong
   2. *Input* **kode buku** tidak boleh mengandung angka **(Hint : gunakan isdigit)**

3. *Input* **harga buku** tidak boleh kurang dari 0
4. **Status buku** akan secara otomatis terisi dengan tulisan "*Standard Price*"

2) Tampil Data (10 poin)

Menu untuk menampilkan seluruh data yang ada pada *array*. Menu ini **tidak bisa diakses jika data pada *array* masih kosong.** Untuk contoh *output* dari tampil data bisa dilihat pada gambar 2.

```
        Buku [1]
Judul   : Harry Pottah
Kode    : HRRRH
Harga   : Rp250000.00
Status  : Standard Price


        Buku [2]
Judul   : The Song of Fire and Ice
Kode    : TSOFI
Harga   : Rp300000.00
Status  : Standard Price


        Buku [3]
Judul   : Jenengmu
Kode    : JNGMU
Harga   : Rp1000000.00
Status  : Standard Price
```

Gambar 2. Contoh Output Tampil Data

3) Cek Palindrom (70 poin)

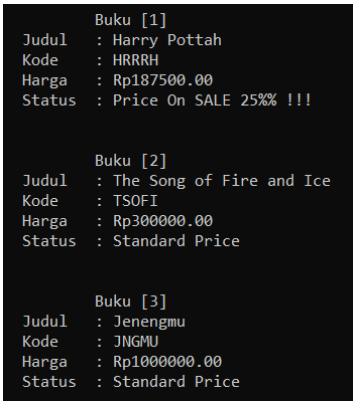
Menu untuk memeriksa seluruh data inputan user, spesifiknya di bagian kode buku. Menu ini juga **tidak bisa diakses jika *array* / data masih kosong**. Untuk melakukan pengecekan Palindrom**, konsep *stack* seperti *push* dan *pop* akan digunakan**. *Char* pada *string* kode akan di-*push* satu per satu ke dalam *stack*. Kemudian, akan dilakukan *pop* pada *stack* tersebut dan hasil *pop* tersebut akan dibandingkan dengan *string* awal. Jika sampai indeks akhir **tidak ditemukan perbedaan**, maka akan melakukan ***return True***. Namun, **jika ditemukan perbedaan**, maka program akan langsung melakukan

return False. Untuk mempermudah dalam membayangkan logika **Palindrom** tersebut, perhatikan alur ilustrasi di bawah ini :

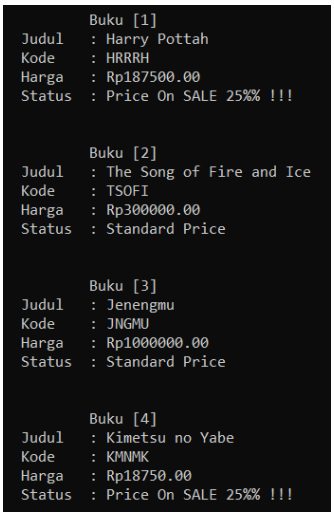
1. *Input* kode : HRRRH
    a. "H" di-*push* ke dalam *stack*, isi dari *stack* sekarang [H]
    b. "R" di-*push* ke dalam *stack*, isi dari *stack* sekarang [H, R]
    c. "R" di-*push* ke dalam *stack*, isi dari *stack* sekarang [H, R, R]
    d. "R" di-*push* ke dalam *stack*, isi dari *stack* sekarang [H, R, R, R]
    e. "H" di-*push* ke dalam *stack*, isi dari *stack* sekarang [H, R, R, R, H]
    f. Dikarenakan *string* sudah mencapai indeks terakhir, maka perulangan untuk memasukkan data dari *string* ke *stack* selesai.
    g. Dilakukan perulangan lagi untuk **membandingkan antara hasil *pop* dari *stack* dengan *string* awal** *input*-an pengguna.
    h. Perulangan akan terus berjalan **selama hasil *pop* dan *char* indeks terkait sama**.
    i. Jika **tidak ada yang berbeda hingga *stack* kosong**, maka program akan melakukan *return* ***True*.**
2. *Input* kode : JNGMU
    a. "J" di-*push* ke dalam *stack*, isi dari *stack* sekarang [J]
    b. "N" di-*push* ke dalam *stack*, isi dari *stack* sekarang [J, N]
    c. "G" di-*push* ke dalam *stack*, isi dari *stack* sekarang [J, N, G]
    d. "M" di-*push* ke dalam *stack*, isi dari *stack* sekarang [J, N, G, M]
    e. "U" di-*push* ke dalam *stack*, isi dari *stack* sekarang [J, N, G, M, U]
    f. Dikarenakan *string* sudah mencapai indeks terakhir, maka perulangan untuk memasukkan data dari *string* ke *stack* selesai.
    g. Dilakukan perulangan lagi untuk **membandingkan antara hasil *pop* dari *stack* dengan *string* awal** *input*-an pengguna.
    h. Jika **ditemukan perbedaan selama membandingkan**, maka **perulangan langsung diberhentikan** dan program akan melakukan *return* ***False*.**

Jika kode buku tersebut **memiliki sifat palindrom**, maka **harga buku** akan **secara otomatis** diubah dengan **diskon 25% (harga awal – (harga awal * 25%))** dan **status**

akan secara otomatis diubah menjadi **"*Price on Sale 25%!!!*"**. Buku yang **sudah dicek dan diberikan diskon 25% tidak akan terpotong diskon lagi** jika pengguna mengakses menu ini kembali. Untuk lebih jelasnya bisa lihat pada gambar 3 dan gambar 4



Gambar 3. Output Tampil Data Setelah Mengakses Menu 3



Gambar 4. Output Tampil Data Setelah Menambah Data dan Mengakses Kembali Menu 3

Bonus (UGD +10)

1) Penghitungan Operasi Prefix

   Buatlah sebuah menu tambahan untuk menghitung hasil akhir dari suatu operasi prefix. Menu ini akan meminta *input* kepada pengguna berupa operasi dalam bentuk prefix. Setelah pengguna melakukan *input*, program akan langsung mengeluarkan hasil dari operasi tersebut. Ada beberapa ketentuan untuk error handling *input* pengguna, seperti:

   1. Operasi tidak boleh kosong.

2. Operasi tidak boleh lebih dari 10 karakter.

Infix	Prefix
A+B	+AB
A+B-C	-+ABC
(A+B)*(C-D)	*+AB-CD

Gambar 5. Contoh Operasi Infix ke Prefix

```
[1]. Operasi Prefix [Bonus]
[0]. EXIT
>>> 1

Masukkan Operasi Prefix : *+23-54

Hasil = 5
```

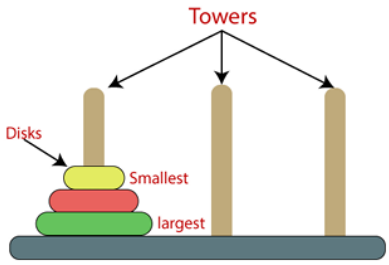
Gambar 6. Contoh Output dalam Program

***HINT* : Gunakan –'0' untuk membantu saat pertama kali menyimpan data dari char ke int**

```
operasi[i] - '0'
```

Gambar 7. Hint Bonus

TUGAS



Gambar 8. Ilustrasi Permainan Tower Hanoi

Edwardy sangat mencintai permainan Menara Hanoi. Maka dari itu Edwardy berencana untuk membuat program tersebut untuk mengisi waktu luangnya. Berikut ketentuan dari program tersebut :

1. Terdapat 3 menara (menara A, menara B dan menara C) yang dapat menyimpan maksimal 3 cakram (gunakan integer sebagai permisalan cakram).

```
        | a |   | b |   | c |

        | 1 |   | 0 |   | 0 |
        | 2 |   | 0 |   | 0 |
        | 3 |   | 0 |   | 0 |
[*] Input 'd' Untuk Keluar

Masukkan angka Tower Yang Hendak Dipindah      :
```

Gambar 9. Visualisasi Tower of Hanoi dalam Program

   Seperti yang bisa dilihat pada gambar 7, cakram dalam menara divisualisasikan dengan integer. Cakram tersebut bernilai 1, 2, dan 3 yang secara langsung juga memberikan besar nilai kepada cakram-cakram tersebut.

2. Cakram dapat dipindahkan dari satu menara ke menara lain dengan memilih menara berisi cakram yang ingin dipindah dan menara tujuannya. Cakram hanya dapat dipindahkan ke menara yang kosong, atau cakram hanya dapat dipindahkan ke menara lain yang memiliki cakram paling atas bernilai lebih besar dari cakram yang akan dipindahkan (cakram yang lebih kecil harus diposisikan paling atas).

Menara Hanoi / *Tower of Hanoi* adalah sebuah permainan matematis atau teka-teki. Permainan ini terdiri dari tiga menara dan sejumlah cakram dengan ukuran berbeda-beda yang bisa

dimasukkan ke menara mana saja. Permainan dimulai dengan cakram-cakram yang tertumpuk rapi berurutan berdasarkan ukurannya dalam salah satu menara, cakram terkecil diletakkan teratas, sehingga membentuk kerucut.

Tujuan dari teka-teki ini adalah untuk memindahkan seluruh tumpukan ke menara yang lain, mengikuti aturan berikut :

- Hanya satu cakram yang boleh dipindahkan dalam satu waktu.
- Setiap perpindahan berupa pengambilan cakram teratas dari satu menara dan memasukkannya ke menara lain, di atas cakram lain yang mungkin sudah ada di menara tersebut.
- Tidak boleh meletakkan cakram di atas cakram lain yang lebih kecil.

KETENTUAN PENGERJAAN

UGD

1. Project Dev C wajib dipisah menjadi 3 bagian (header.h, main.c, source.c)
2. Pastikan Code dapat di compile dengan lancar pada saat mengumpul
3. Harus memakai Stack dalam code (jika tidak memakai akan langsung diberikan nilai 0)
4. Bonus hanya dapat diklaim jika tidak bertanya kepada asisten (kecuali mengenai kejelasan soal) dan telah berhasil mengerjakan soal UGD dengan benar
5. Selesai UGD + Semua Bonus selama pratikum, nilai UGD 110
6. Dilarang membuka jawaban dari praktikan kelas lain ataupun bekerja sama dengan praktikan lain selama praktikum berlangsung (jika ketahuan, nilai dalam Spreadsheet akan langsung dimerahkan / diberikan pelanggaran plagiasi)
7. Ketentuan lain dalam Spreadsheet juga berlaku

TUGAS

1. File UGD dan Tugas tidak perlu dipisah
2. Tindakan plagiasi tidak akan diberikan toleransi
3. Ketentuan lain dalam Spreadsheet juga berlaku

LAPORAN

1. Selesaikan Tugas terlebih dahulu
2. Laporan berisi penjelasan singkat mengenai alur program UGD dan Tugas
3. Maksimal 3 halaman tanpa cover
4. Berikan identitas berupa nama, npm, dan kelas
5. Ketentuan lain bisa dilihat pada Spreadsheet

FORMAT PENGUMPULAN

Unguided	Unguided + Bonus	Tugas	Laporan
UGD3_Y_XXXXX.zip	UGD3_Y_XXXXX_BONUS.zip	TGS3_Y_XXXXX.zip	LAP3_Y_XXXXX.pdf

- Y = Kelas
- X = 5 digit NPM terakhir