ADOBO CODE v1.0

***Project Manager*:** Julius Jireh B. Vega

Assistant Project Manager: Aleli S. Legaspi

Members: Jan Charles M. Adona, Miles Lawrence R. Basnillo, Maru Gabriel S. Baul, Emmanuel B. Constantino Jr., Adriell Dane C. de Guzman, Adrian Carlos A. de Vera, Kidd Jason C. Evangelista, Joab C. Fernando, Nurfitra A. Jafaar, Christine Mae H. Juruena, Renee Arianne F. Lat, Juan Carlo M. Mangaliag, Darius M. Orias, Irvin Kean Paulus T. Paderes, Gio Joshua B. Peralta, JC Carlo D.G. Quintos, Paul Joshua H. Robles, Aron John S. Vibar

PAGGAWA NG *FILE*

".adobo" ang karugtong na pangalan ng *file* ng adobo code. '\n' o '\r' o '\n\r' ang pantapos sa bawat kowd. Dapat ang isang adobo *file* ay may `SIMULA()` na *function* at ito ay dapat tapusin gamit ang `WAKAS`. Sa `SIMULA()` unang pupunta ang *interpreter* sa pagpapatakbo ng kowd. Pwedeng hindi malaking titik ang gamitin sa pagsusulat ng mga salitang may kahulugan sa adobo code pero ang pangalan ng *variables* at pangalan ng *functions* ay kailangang gumamit ng striktong malalaki at maliliit na letra (*case-sensitive*).

KOMENTO

Ang mga komento ang nilalagay sa kowd pero hindi papansinin ng *interpreter*. KOMENTO ang gagamitin para magkomento ng isang linya lamang.

Syntax:

```
KOMENTO <komento>
```

Ito naman ang *syntax* para sa mga komentong lalampas pa sa isang linya:

```
MGA KOMENTO:
     <komento>
DULO NG KOMENTO
```

PANGUNAHING GINAGAWA (*MAIN FUNCTION*)

Ang pangunahing ginagawa ay isang natatanging *function* sapagkat ito ang hinahanap sa isang kodigo upang mapatakbo lahat ng kowd sa loob nito. Ito ay ginawa upang hindi maduming tingnan ang kodigo kapag marami na ang *functions* sa kowd.

Syntax:

```
SIMULA()
     <grupo ng mga gawain>
WAKAS
```

Kapag tinawag ang IBALIK sa SIMULA, ito ang magpapapigil sa pagpapatakbo ng ating palatuntunan.

VARIABLES

Primitibong Datos

Ang mga primitibong datos ay ang mga datos na hindi na pwedeng putulin pa sa mas maliit na datos. Ito ang mga sumusunod na primitibong dato:

1. BILANG

Ito ang magsisilbing representasyon ng mga numerong walang putal (*whole numbers*). 0 ang kaniyang halaga kapag walang nilagay na halaga pagkatapos ihayag.

2. NUMERO

Ito ang magsisilbing representasyon ng mga totoong numero. Ito ay may *decimal point* pero hindi kailangan na meron ito. 0.0 ang kaniyang halaga kapag walang nilagay na halaga pagkatapos ihayag.

3. SIMBOLO

Ito ang representasyon ng mga iba't ibang simbolo at letra. Dapat ang halaga ng SIMBOLO ay pinagigitnaan ng kudlit (' ') kung hindi ito ay magbibigay ng *error*. Hindi pwede maglagay ng halagang BILANG sa SIMBOLO kung hindi ito ay magbibigay ng *warning*. Puwang (*space)* ang halaga kapag walang inihayag na halaga. *ASCII character set* ang gagamitin para mapalit ang BILANG papuntang SIMBOLO.

4. SALITA

Ito ay representasyon ng mga grupo ng simbolo. Kahit SALITA ang tawag, ito ay pwedeng magkaroon ng puwang. Ang halaga ng SALITA ay dapat pinagigitnaan ng panipi (" ") kung hindi ito ay magbibigay ng *error*. Isang puwang *(space*) ang halaga kapag walang inihayag na halaga.

5. SAGOT

Ito ay ginagamit para sa mga kundisyon. Ito ay pwedeng lamang maging TAMA o MALI. Kapag walang nilagay na halaga ang isang SAGOT na *variable*, ito ay may halagang TAMA.

KOLEKSYON

Ang KOLEKSYON ay isang uri ng datos na pwedeng makahawak ng maraming primitibong datos. Ang paglalagay at pagtatanggal ng halaga ay laging ginagawa sa dulo ng KOLEKSYON.

Iba't ibang *functions* sa KOLEKSYON

May iba't ibang *functions* na ginawa para sa KOLEKSYON. Ito ay pwedeng gamitin kaagad at hindi na ito kailangan ipahayag sa kodigo. Para magamit ito, gagamit tayo ng ganitong palaugnayan:

```
<pangalan ng function>([<argumento1>, <argumento2>, ...])
```

Ang mga sumusunod ay ang mga iba't ibang *functions* sa KOLEKSYON:

1. LAGAY(<KOLEKSYON na *variable*>, <halaga>)

   Hinahanap nito ang dulo ng KOLEKSYON at ilalagay ang <halaga> doon.

2. TANGGAL(<KOLEKSYON na *variable*>)

   Hahanapin nito ang pinakadulong <halaga> ng KOLEKSYON at tatanggalin ito sa KOLEKSYON.

3. KUHA(<KOLEKSYON na *variable*>, <bilang>)

   Kukunin nito ang <halaga> sa ika-<bilang> sa KOLEKSYON. Kapag ang <bilang> ay sumobra sa totoong bilang ng KOLEKSYON, titigil ang palatuntunan at magsasabi ito ng mali sa palatuntunan.

   One-indexing ang gagamitin. Ibig-sabihin, ang pinakaunang bagay sa KOLKESYON ay may bilang na 1.

4. PALIT(<KOLEKSYON na *variable*>, <bilang>, <halaga>)

   Papalitan nito ang <halaga> na nandoon sa ika-<bilang> sa KOLEKSYON ng bagong <halaga>.

5. HANAP(<KOLEKSYON na *variable*>, <halaga>)

   Hahanapin ang <halaga> sa KOLEKSYON. Kung nakita ang <halaga>, ibabalik ng gawain ang <bilang> ng <halaga>. Kung hindi nahanap, magbabalik ng MALI.

6. ILAN(<KOLEKSYON na *variable*>)

   Ibabalik nito ang bilang ng mga <halaga> sa KOLEKSYON.

Iba't ibang *functions* sa SALITA

Ang SALITA ay isang primitibong uri ng datos pero may *functions* din na ginawa para sa SALITA tulad lamang din ng KOLEKSYON. Ito ang mga sumusunod

1. DUGSONG(<SALITA na *variable1*>, <SALITA na *variable2*>)

   Pagsasamahin nito ang dalawang SALITA. Magsisimula ang <SALITA na *variable2*> sa dulo ng <SALITA na *variable1*>.

2. HATIIN(<SALITA na *variable*>, <SIMBOLO na *variable*>)

   Hahatiin nito ang isang SALITA upang makagawa ng KOLEKSYON NG SIMBOLO. Gagamitin nito ang SIMBOLO para hatiin ang SALITA.

Halimbawa:

```
SALITA word NA MAY "kakarampot"
HATIIN(word, "") <-- magbabalik ng  ["k", "a", "k" …]
HATIIN(word, "r") <-- magbabalik ng ["kaka", "ampot"]
```

3. BUUIN(<KOLEKSYON NG SALITA na *variable*>)

Ibubuo nito ang SALITA gamit ang magkakahiwalay na SIMBOLO sa KOLEKSYON NG SIMBOLO.

Pagpapahayag ng *variable* (*variable declaration*)

Kailangan munang ihayag ang isang *variable* bago ito magamit. Para makapagpahayag ng isang *variable*, kailangan sabihin muna kung ano ang uri ng datos. Ang palaugnayan para makapagpahayag ng isang *variable* ay ang susunod:

<uri ng datos> <pangalan ng variable> [NA MAY <halaga>]

Sa KOLEKSYON, medyo iba ang *syntax* para magpahayag ng *variable*. Ito ay sisimulan sa KOLEKSYON at susundan ng NG <uri ng datos> na magpapatakda ng uri ng datos na pwedeng ilagay lang sa loob ng KOLEKSYON. Kapag ibang uri ng datos ang nilagay sa KOLEKSYON, ito ay magbabalik ng *error*.

Syntax sa paghayag ng KOLEKSYON:

KOLEKSYON NG <uri ng datos> <pangalan ng *variable*> [NA MAY <halaga1>, <halaga2>, … ATSAKA, <halagaN>]

Ang salitang ATSAKA ay ginagamit lamang kung ang mga ilalagay na halaga ay mas marami pa sa isa. Ilalagay lang ang ATSAKA bago sabihin ang pinakahuling halaga na ilalagay sa KOLEKSYON.

Ang halimbawa ng paghayag ng KOLEKSYON na may halaga:

KOLEKSYON NG SALITA bayong NA MAY "isda"

KOLEKSYON NG SALITA bayong NA MAY "isda", ATSAKA "baka"

KOLEKSYON NG SALITA bayong NA MAY "isda", "baboy", "baka", ATSAKA "kabayo"

Sa pagpangalan ng *variable*, pwede ito magsimula sa letra o sa salungguhitan. Pagkatapos pwede itong sundang ng letra, numero, salungguhitan, o gitling.

Ang halimbawa sa baba ay nagpapahayag ng *variable* na walang ideneklarang halaga.

BILANG y

Ang halimbawa sa taas ay nagpapahayag ng *variable* na may halagang 5.

BILANG x NA MAY 5

Mga Operasyon sa Mga *VARIABLES*

1. Palatuusan (*Arithmetics*)

Ang mga palatuusan na gagamitin ay parehas lang sa kanilang katumbas nilang simbolo. + sa pagdadagdag, − sa pagbabawas, / sa mala-matematikong paghahati, // sa mala-bilang na paghahati, * sa pagdarami ngunit kailangan ang RESULTA NG bago ipahayag ang mga palatuusan na gagawin.

Ang mala-matematikong paghahati(/) ay maglalabas ng NUMERO kahit hindi NUMERO ang nasa panakda at pamanagi. Ang mala-bilang na paghahati (//) ay palaging maglalabas ng BILANG pero dapat walang putal ang mga halaga sa panakda at pamanagi kung hindi ito ay maglalabas ng *error*. Pareho silang maglalabas ng *run-time error* kapag naging 0 ang pamanagi.

2. Paghahambing (*Comparison*)

'AY MAS MALAKI SA' ang magsasabi kung ang halaga sa kaliwa ay mas malaki sa halaga na nasa kanan. 'AY MAS MALIIT SA' ang magsasabi kung ang halaga sa kanan ay mas malaki sa halaga sa kaliwa. 'AY PAREHO SA' ang magsasabi kung pantay lang ang mga *operand* sa kaliwa at sa kanan.

'AY MAS MALAKI O PAREHO SA' ang ginagamit para sa pinagsamang 'AY MAS MALAKI SA' at 'AY PAREHO SA'. 'AY MAS MALIIT O PAREHO SA' ang ginagamit para sa pinagsamang 'AY MAS MALIIT SA' at sa 'AY PAREHO SA'. 'AY HINDI PAREHO SA' ang magsasabi kung hindi pantay ang mga *operand* sa kaliwa at sa kanan.

Kapag SIMBOLO o SALITA ang inihahambing, titingnan nito ang alpabetikong pag-aayos ng mga simbolo. Mas malaki ang halaga kapag mas malapit sa Z at mas maliit ang halaga kapag mas malapit sa A.

3. Pag-uugnay ng mga pinaghahambing (*Logical*)

Para mapagsama ang mga pinaghahambing sa isang utos, pwedeng gumamit ng mga pang-ugnay ng mga ppinaghahambing. 'AT' ang magsasabi kung ang mga kundisyon ay parehas na TAMA kung hindi ito ay magbabalik ng MALI. 'O' ang magsasabi kung kahit isa sa mga kundisyon ay TAMA kung hindi ito ay magbabalik ng mali.

<u>PAGLILIMBAG AT PAGKUHA NG IMPORMASYON</u>

Paglalabas o Paglilimbag ng *Data*

Para maglimbag o maglabas ng mga salita o mga <halaga> sa *console*, gagamitin ang mga salitang ISULAT ANG.

Syntax:

```
ISULAT [ANG] <halaga1>[, [<halaga2>, …]
```

Kung KOLEKSYON ang nilagay sa <halaga>, ipapakita nito lahat ng <halagang> hinahawak ng isang KOLEKSYON.

Ang isang halimbawa nito ay kapag may ganitong KOLEKSYON:

```
KOLEKSYON NG SALITA miyembro NA MAY "Miles", "Aron", ATSAKA "PJ"
```

At ginawa ito:

```
ISULAT ANG miyembro
```

Ang ilalabas nito sa *console* ay ang mga sumusunod:

```
["Miles", "Aron", "PJ"]
```

Escape Characters

Para makapaglabas ng mga simbolo na nareserba sa ibang ginagawa, gagamitin ang *backslash* para dito. '\linya' ang gagamitin para maglabas ng bagong linya ang paglalabas. Ito ang halimbawa ng pagkakagamit ng *escape characters*:

```
ISULAT ANG "Ang pangalan ko ay ", pangalan, ".\linya"
```

Pagpapasok o Pagkukuha ng *Data*

Para humingi ng halaga na ipapasok ng gumagamit, gagamitin ang mga salitang HINGI NG. Ito ang palaugnayan ng HINGI NG:

```
HINGI [NG] <pangalan ng variable>
```

Hindi pwedeng KOLEKSYON na *variable* na ilalagay sa HINGI NG. Kailangan muna itong ilagay sa ibang *variable* saka lang gagamitin ang LAGAY o KUHA ng KOLEKSYON para makapaglagay ng HALAGA gamit ang HINGI NG.

Halimbawa:

```
BILANG x NA MAY 5
KOLEKSYON NG BILANG data
~~HINGI NG data <--hindi pwede ito~~
HINGI NG x
LAGAY(data, x) <-- pwede ito
```

<u>KUNDISYON</u>

Para gumawa ng mga utos na may kundisyon, kailangang gamitin ang KUNG, O KUNG, at KUNG HINDI. Ito ay gagamit ng mga sumusunod na palaugnayan:

```
KUNG <pinaghahambing>
     <grupo ng gawain>
[O KUNG <pinaghahambing>
     <grupo ng gawain>
```

```
[O KUNG <pinaghahambing>
     <grupo ng gawain>
...]
[KUNG HINDI
     <grupo ng gawain>]
DULO NG KUNG
```

Sa pagtitingin ng mga gawain na gagamitin, titingnan nito ang <pinaghahambing> sa KUNG. Kapag ito ay TAMA, papasok ito sa mga <grupo ng gawain> sa loob ng KUNG.

Kapag mali ang <pinaghahambing> ng KUNG, saka lang ito papasok sa susunod na O KUNG, kung meron man. Ganoon ulit ang proseso, titingnan niya kung TAMA ang <pinaghahambing> sa O KUNG. Kung TAMA, ito ay papasok sa mga <grupo ng gawain>.

Kapag lahat ng <pinaghahambing> sa KUNG at O KUNG ay MALI, saka lang ito papasok sa mga <grupo ng utos> sa loob ng KUNG HINDI.

MGA IKOT

Ang ikot ay ang pa-ulit ulit na pagsasagawa ng mga gawain. Ito ay may tatlong parte: kung saan magsisimula <pagpapahayag>, kung hanggang kailan ito tatakbo <kundisyon>, at kung anong pagbabago ang mangyayari <pagbabago>. HABANG ang gagamitin para makapagpahayag ng ikot.

Syntax:

```
HABANG
     <pagpapahayag>
     <kundisyon>
     <pagbabago>
GAWIN
     <grupo ng utos>
DULO NG HABANG
```

Dapat ang mga ipinahayag nga *variable/s* ay hindi pa dapat naipapahayag pa. Ang mga naipahayag na variables sa <pagpapahayag> ay hindi pwedeng gamitin sa ibang parte ng palatuntunan.

Ang halimbawa ng HABANG na gumagamit ng isang *variable* sa ikot:

```
HABANG
     BILANG i NA MAY 0
     i AY MAS MALIIT SA 10
     ILGAYA SA i ANG RESULTA NG i+1
```

```
GAWIN
      ISULAT ANG i
DULO NG HABANG
```

Ang halimbawa ng HABANG na gumagamit ng dalawang *variable* sa ikot:

```
HABANG
      BILANG i NA MAY 0, BILANG j NA MAY 0
      i AY MAS MALIIT SA 10
      ILAGAY SA i ANG RESULTA NG i+1, ILAGAY SA j ANG RESULTA NG j+2
GAWIN
      ISULAT ANG i, " ", j
DULO NG HABANG
```

Para naman sa hindi matapos na pag-iikot ito ang gagamiting na palaugnayan:

```
HABANG
GAWIN
      <grupo ng utos>
DULO NG HABANG
```

Pwedeng gamiting ang TIGIL para pigilan ang ikot kahit TAMA pa ang kundisyon. Pwedeng gamitin ang TULOY para laktawan ang mga utos at pumunta kaagad sa susunod na ikot. ngunit tandaan na ito ay dadaan sa <pagbabago> bago magsimula ang susunod na ikot

MGA *FUNCTIONS*

Paggawa ng *Functions*

Ang *function* ay grupo ng mga gawain na maaring gawin nang isahan upang hindi ito magpaulit-ulit sa kowd. Ang kailangan para makagawa ng isang *function* ay ang pangalan ng *function* at ang listahan ng mga parametro. Ito ay tinatapos lagi ng WAKAS.

Syntax:

```
<pangalan ng ginagawa>([<paramtero1>, <parametro2>, …])
      <grupo ng mga gawain>
WAKAS
```

Sa pagpapangalan ng ginagawa, ito ay pwedeng magsimula sa letra o sa salungguhitian(_). Pagkatapos, pwede itong sundang ng letra, numero, salungguhitan, o gitling(-) **ngunit** hindi pwede ang salitang SIMULA sapagkat ito ay nakareserba para sa pangunahing ginagawa.

Para tawagin ang isang ginagawa, gagamitin natin ang TAWAGIN ANG. Ang ginagawa nito ay pinapahayag niya na magtatawag tayo ng isang *function*.

Syntax:

```
TAWAGIN ANG <pangalan ng ginagawa> ([<argumento1>, <argumento2>,...])
```

Ang mga ginagawa ay pwede ring magbalik ng halaga pagkatapos gawin ang mga utos. Kumbaga ito ung papalit na halaga sa kaniya kapag tinawag. Para makapagbalik ng halaga, gagamitin ang salitang `IBALIK`.

Syntax:

```
IBALIK [ANG] <halaga>
```

Ang `ANG` ay maaaring hindi na ilagay.

Kapag nakita na ang `IBALIK`, hindi na patatakbuhin ang ibang utos dahil bumalik na siya sa isang ginagawa na tinawag bago ito.

Ito ang halimbawa ng paggawa ng *function*:

```
SIMULA()
    BILANG A NA MAY idagag(5, 3)
    ISULAT ANG A
WAKAS

idagdag(BILANG X, BILANG Y)
    IBALIK ANG RESULTA NG X+Y
WAKAS
```

Ang kowd sa taas ay magpapakita ng "`8`" sapagkat ang tinawag ng `SIMULA` ang `idagdag` na may argumentong `5` at `3`. Pagkapasa ng `5` at `3` sa `idagdag`, ang `X` ay kinuha ang halagang `5` at ang `Y` ay kinuha ang halagang `3`. Pagkatapos, pinagsama ang halaga ng `X` at `Y` at pagkatapos ay ibinalik ang resulta. Kaya ang nalagay na halaga kay `A` ay `8`. Pagkatapos nito, sinulat ang `A` sa *terminal* kaya ito ay nagpakita ng `8`.

Ang mga *functions* ay dapat ilagay pagkatapos ng `SIMULA`. Ito ay magbibigay ng *error* kung hindi ito nasunod.