

Begin Modern Programming

with

C
O
D
I
T
Y
A

Pyi Soe

အခန်း ၁

စက်ရုပ်ကားရဲလိုဖြင့် ပရီဂရမ်းမင်းမိတ်ဆက်

ကွန်ပျူတာတွေဟာ သက်မဲ့ စက်ပစ္စည်းတွေပါပဲ။ ကားတို့ လေယာဉ်တို့နဲ့ မတူတာက ကွန်ပျူတာတွေဟာ စက်ချဉ်းသက်သက် ဘာအစွမ်းမှ မယ်မယ်ရရ မရှိဘူး။ ဒါပေမဲ့ ဆောင်ရွက်လိုတဲ့ ကိစ္စအဝေဝအတွက် ပရီဂရမ်းမျိုးမျိုး ထည့်ပေးလိုက်တဲ့ အခါမာ သူ့အစွမ်းက အတိုင်းအဆမဲ့ပဲ။ နေရာမျိုးစံ၊ နယ်ပါယ်မျိုးစံ မှာ အကူးအညီပေးနိုင်တဲ့ စွယ်စုံသုံး ပစ္စည်းတစ်ခုဖြစ်သွားတယ်။ ဂိတ်သံစဉ်တွေကို ဖွင့်ပေးနိုင်သလို အသံလည်းသွင်းပေးနိုင်တယ်။ ရုပ်ရှင်တည်းဖြတ် လုပ်ချင်တာလား။ ပြဿနာမရှိဘူး၊ ကူညီပေးနိုင်တယ်။ နျှော လီးယား ပါတ်ပေါင်းဖို့တွေကို စီမံခိုင်သလို မောင်းသူမဲ့ အုံပျုံတွေကိုလည်း ပဲထိန်းပေးနိုင်တယ်။

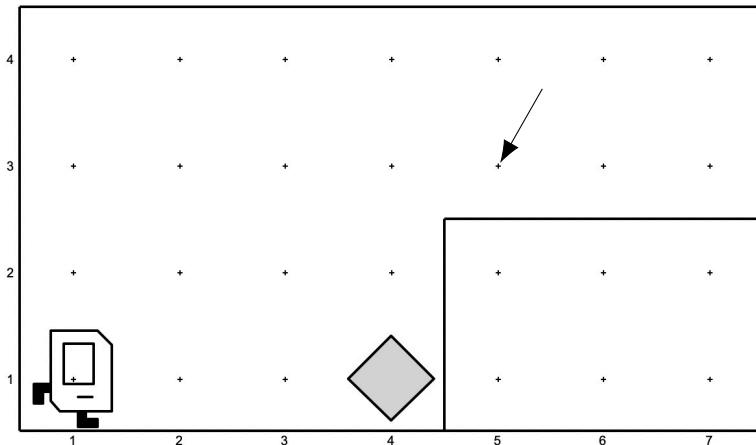
ကျွန်တော်တို့တွေ နိစ္စဓာဝ အသုံးပြုနောက်တဲ့ ကား၊ စမတ်ဖုန်း၊ လက်ပါတ်နာရီ၊ မိုက်ခရှိဝေဖွံ့ဖို့ အဝတ်လျှော့စက် စတဲ့ စက်ပစ္စည်း အမျိုးမျိုးဟာလည်း ကွန်ပျူတာတွေနဲ့ မကင်းပြန်ပါဘူး။ “ကွန်ပျူတာနည်းပညာ အကူးအညီမပါတဲ့ အတိုင်းဆက်သံစုံတွေမှာ မရှိဘူး” လို့ ဆိုနိုင်ပါတယ်။

တစ်ချက်တစ်ချက် ရိုက်ခတ်လိုက်တဲ့ ကွန်ပျူတာနည်းပညာ လိုင်းလုံးကြီးတွေဟာ ကမ္မာတစ်စုံမှုးလုံး ပုံစံပြောင်းသွားလောက်အောင် အဟုန်ပြင်းထန်လာတယ်။ ဘီလီယံချွဲတဲ့ လူတွေ ဆီရှယ်မီဒီယာတွေပေါ်က နေ ရုပ်သံတွေနဲ့ ချိတ်ဆက်ပြောဆိုသွေးလို့ ရစေတာဟာလည်း ကွန်ပျူတာစနစ်တွေပါပဲ။ Artificial Intelligence (AI) နည်းပညာကြောင့် သက်ရှိတွေမှာပဲတွေ့ရတဲ့ ညာတ်ရည်မျိုးကို ကွန်ပျူတာတွေမှာလည်း တွေ့လာရပါပြီ။ သံချွဲပုံစံတွေ ဖြေရှင်းခြင်း၊ စစ်တုရင်ထိုးခြင်း စတဲ့ ကိစ္စမျိုးတွေအပြင် ပန်းချီဆွဲခြင်း၊ ကဗျာရေးစပ်ခြင်း၊ သီချင်းရေးဖွံ့ခြင်း ကဲ့သို့ အနုပညာဖန်တီးမှုတွေကိုပါ AI က လုပ်ဆောင်ပေးနိုင်ပါတယ်။ နှစ်ဆယ်တစ်ရာစုံ အထူးမြှားဆုံး AI နည်းပညာလိုင်းဟာ အရှိန်အဟုန်ပြင်းပြင်း ရိုတ်ခတ်ဖို့ အားယူစွဲပြုနေပါပြီ။

‘ကွန်ပျူတာ’ လိုပြောတဲ့ အခါ စက်ပစ္စည်းသက်သက် မဟုတ်ဘဲ ကွန်ပျူတာမှုတ်ညာတ်တဲ့ ပရီဂရမ်တွေလည်း ပါဝင်တယ်ဆိုတာ သတိချုပ်ရပါမယ်။ ကွန်ပျူတာတွေ တစ်စုံတစ်ရာ စွမ်းဆောင်နိုင်စေတဲ့ ပရီဂရမ်တွေ ရေးတဲ့ အလုပ်ကို ပရီဂရမ်းမင်း (Programming) လို့ခေါ်တယ်။

၁.၁ စက်ရုပ် ကားရဲလို

ပရီဂရမ်းမင်းဆိုတာ ဘယ်လိုမျိုးလဲ သဘောပေါက်အောင် စာတွေတစ်သီဥပါးရေး ရှင်းပြတာထက် ပရီဂရမ်လေးတွေ လက်တွေ၊ ရေးကြည့်လိုက်တာ ပိုပြီးထိရောက်ပါတယ်။ ဒါကြောင့် စက်ရုပ်ကားရဲလိုကို ပရီဂရမ်လေးတွေရေးပြီး အလုပ်တွေခုံးကြည့်ကြမယ်။ ပုံ (၁.၁) မှာ တွေ့ရတာက ကားရဲလို ရောက်ရှိနေတဲ့ နဗုံနာကမ္မာတစ်ခုပါ။ မီးခါးရောင် မှန်ကူကူပုံလေးကို ဘိပါ (beeper) လို့ ခေါ်တယ်။ အဲဒီဘိပါကို မြားပြထားတဲ့ နေရာကို ရွှေခိုင်းချင်တယ်။ မျှေားမည်းအထူးတွေက နံရံတွေပါ။ ကားရဲလိုကို ကိစ္စတစ်ခု ဆောင်ရွက်စေ



ပုံ ၁.၁ ခက်ရုပ်လေး ကားရဲလ်

ချင်တဲ့အခါ အခြေခံ ကားရဲလ်ကွန်မန်းတွေကို အသုံးပြုရပါတယ်။ ကွန်မန်းတွေကို နှုတ်နှုံးပြောပြီး ခိုင်းရတာ မဟုတ်ဘဲ ပရိုကရမ်းပြီး ခိုင်းရတာပါ။ ကားရဲလ်နားလည်တဲ့ ကွန်မန်းတွေကို ကြည့်ကြရအောင်။

ကားရဲလ်ကွန်မန်းများ

မဖြစ်မနေ သိထားရမဲ့ အခြေခံ ကားရဲလ်ကွန်မန်း လေးခုပဲ ရှိတယ်။ move, turn_left, put_beeper နဲ့ pick_beeper တို့ဖြစ်တယ်။ အခြား ကားရဲလ်ကွန်မန်း တွေလည်း ရှိပါသေးတယ်။ ဒါပေမဲ့ ကားရဲလ် ပရိုကရမ်းမင်း စလေ့လာဖို့ ဒီလေးခုနဲ့ပဲ လုံလောက်ပါပြီ။

move ကွန်မန်းက ကားရဲလ်ကို ရှေ့တစ်ကွန်နာကို ရွှေ့ခိုင်းတာ။ ကားရဲလ်ကမ္ဘာထဲမှာ တစ်ခုနဲ့တစ်ခု အကွာအဝေးတူ ခြားထားတဲ့ အတန်းလိုက် အတန်းလိုက် အစက်ကလေးတွေဟာ ကွန်နာ (corner) တွေ ဖြစ်တယ်။ ကမ္ဘာကို ပျော်မည်းမည် အထူး နံရုံတွေနဲ့ ထောင့်မှန်စတုဂံပဲ ပါတ်လည် ဘောင်ခတ်ထားတယ်။ ကွန်နာတွေကြားမှာလည်း နံရုံတွေရှိနိုင်တယ်။ နူးနာကမ္ဘာမှာ ဘေးတိုက် နံပါတ်စဉ် ၄ နဲ့ ၅ ကြား ထောင်လိုက် နံရုံတစ်ခု၊ အထက်အောက် နံပါတ်စဉ် ၂ နဲ့ ၃ ကြား အလျေားလိုက် နံရုံတစ်ခုကို တွေ့ရှုပါမယ်။ ကွန်နာရှေ့မှာ နံရုံကာနေရင် ကားရဲလ်ကို move ခိုင်းလို့မရပါဘူး။

put_beeper က ကားရဲလ် လက်ရှိ ရှိနေတဲ့ ကွန်နာမှာ ‘ဘိပါတစ်ခုချု’ ထားခိုင်းတာ၊ pick_beeper က ရပ်နေတဲ့ ကွန်နာမှာ ‘ဘိပါတစ်ခုကောက်’ ခိုင်းတာပါ။ ကွန်နာမှာ ဘိပါရှိနေမှ ကောက်ခိုင်းလို့ရမှုပါ။ မရှုရင် ကောက်ခိုင်းလို့ မရှုဘူး။ ဘိပါချိခိုင်းရင်လည်း ကားရဲလ်မှာ ဘိပါရှိမှ ချိခိုင်းလို့ရတယ်။ ကားရဲလ်ကို ဘိပါတွေ လိုသလောက် ဖြည့်ပေးထားတယ်လို့ ယူဆပါ။ turn_left က ‘ဘယ်လှည့်’ ခိုင်းတာ။

ဘိပါကို ဘယ်လိုရွှေ့ခိုင်းမလဲ

ပုံ (၁.၁) အနေအထားကနေ ရေ့ကို သုံးနေရာရွှေ့ ဘိပါကောက်၊ ဘယ်ဘက်လှည့်၊ အပေါ် နှစ်နေရာရွှေ့ ညာဘက်လှည့်၊ ရှေ့တစ်နေရာထပ်ရွှေ့ပြီး ဘိပါချုထားခိုင်းလိုက်ရင် အလုပ်ပြီးသွားပါပြီ။

ကားရဲလ်ကို ညာဘက်လှည့်ခိုင်းဖို့ turn_right ကွန်မန်း မရှိဘူး။ ဒါပေမဲ့ ဘယ်သုံးခါလှည့်တာဟာ ညာဘက်လှည့်တာနဲ့ တူတူပါပဲ။ ဒါကြောင့် ညာဘက်ချင်တဲ့အခါ ဘယ်သုံးခါလှည့်ခိုင်းလို့ရတယ်။

၁.J Meet Karel ပရိုဂရမ်

ပရိုဂရမ် ရေးတယ်ဆိုတာ ကွန်ပျူးတာကို ကိစ္စတစ်ခုခဲ့ အောက်ရှုက်ပေးဖို့ ခိုင်းစေတဲ့ ညွှန်ကြားချက်တွေ ရေးတာပါပဲ။ ဒီလို ညွှန်ကြားချက်တွေကို ပရိုဂရမ်ကုဒ် (program code) လို ခေါ်တယ်။ ပရိုဂရမ်ကုဒ် တွေကို ကွန်ပျူးတာနားလည်တဲ့ programming language တစ်ခုခဲ့ ရေးရတယ်။ ဒီစာအုပ်မှာ အသံးပြုမဲ့ programming language ကတော့ Python ပါ။ Programming language တစ်မျိုးပဲ ရှိတာ မဟုတ်ပါဘူး။ ရာနဲ့ချိပြီး ရှိတာပါ။ လူဘာသာစကားတွေ အမျိုးမျိုးရှိသလိုပဲပေါ့။ Python ဟာ ဒီလို ရာနဲ့ချိတဲ့ထဲက လက်ရှိအသံးအများဆုံး ထိပ်ဆုံးဆယ်ခု ထဲမှာ ပါဝင်တယ်။ Python နဲ့ ဘိပါန္တာခိုင်းတဲ့ ပရိုဂရမ်ကို လေ့လာကြည့်ရအောင်။ ကားရဲလ်နဲ့ ပထမဆုံး မိတ်ဆက်ပေးတဲ့ ပရိုဂရမ်မို့လို ဒီပရိုဂရမ် နံမည်ကို ‘Meet Karel’ လို ခေါ်ပါမယ်။

```
# File: meet_karel.py
# About: This is
from stanfordkarel import *

def main():
    """Karel code goes here!"""
    move()
    move()
    move()
    pick_beeper()
    turn_left()
    move()
    move()

    turn_left()
    turn_left()
    turn_left()

    move()
    put_beeper()
# End of main

if __name__ == "__main__":
    run_karel_program("meet_karel")
```

ဒါဟာ ‘Meet Karel’ ပရိုဂရမ်အတွက် Python နဲ့ရေးထားတဲ့ ပရိုဂရမ်ကုဒ် တွေဖြစ်ပါတယ်။ ‘Python ကုဒ်’ လို အတိုကျိုးပဲ ပြောတာများတယ်။ Python ‘စာ/စကား’ မတတ်ရင် ဒီ ‘Python ကုဒ်’ တွေကိုလည်း နားလည်မှာ မဟုတ်ဘူး။ ဒီတော့ Python ‘စာ/စကား’ အခြေခံက စပြီး လေ့လာဖို့လိုပါမယ်။

ကွဲနဲ့မနဲ့ (Comment)

ပထမဆုံး # သင်္ကာနဲ့ စတဲ့ စာကြောင်းတွေက ကွန်းမန်တွေပါ။ ကွန်းမန်တွေက ကွန်ပျူးတာ အောင်ရှုက်ပေးရမဲ့ ညွှန်ကြားချက်တွေ မဟုတ်ဘူး။ ပရိုဂရမ်ကုဒ်နဲ့ ပါတ်သက်ပြီး ကုဒ် ဖတ်ရှုသူ အတွက် မှတ်ချက်ရေးတာ သို့မဟုတ် ရှင်းပြထားတာပါ။ တနည်းအားဖြင့် ဖတ်ရှုသူ (လူ) ပရိုဂရမ်မှာအတွက် ရည်ရွယ်တာ။ ကွန်ပျူးတာ (စက်) အတွက် ရည်ရွယ်တာ မဟုတ်ဘူး။ ကွန်ပျူးတာက ကုဒ်ထဲက ကွန်းမန်တွေ အားလုံးကို

လစ်လျှောက်မှုပါ။ ဒါပေမဲ့ ပရိုဂရမ်ကုဒ်ကို ဖတ်တဲ့လူ နားလည်ဖို့ အထောက်အကူးဖြစ်စေတဲ့အတွက် ကုန်းမန်ရေးတာကို ပေါ့ပေါ့တန်တန် အရေးမပါသလို သဘောထားလို့ မရပါဘူး။ မိမိရေးတဲ့ ကုဒ်ကို ရှင်းပြနိုင် လိုအပ်ရင် ကုန်းမန်ရေးရပါမယ်။ ရေးသင့်တဲ့ နေရာတွေကိုလည်း မကြာခင်တွေရမှာပါ။

import စတိတ်မန္တ

```
from stanfordkarel import *
```

ကတေသာ အင်ပို့စတိတ်မန် ဖြစ်ပါတယ်။ “stanfordkarel လိုက်ဘရီမ အာလုံးကို ထည့်သင်းပေးပါ” လိုတောင်းဆိုတဲ့ အဓိပ္ပာယ်။ * သက်ဗောက်ကို ‘အားလုံး’ လို့ ယူဆပါ။ stanfordkarel လိုက်ဘရီမှ ကားရဲ့လိုပရိုဂရမ်အတွက် လိုအပ်တာအားလုံး ပါဝင်တယ်။ ဒီလိုက်ဘရီကို အင်ပို့လုပ်ထားမှ ကားရဲ့လိုက်ဘရီမှ တွေ သုံးလို့ရမှာပါ။ သီးခြား ကားရဲ့လိုပရိုဂရမ် တစ်ခုစီတိုင်းအတွက် အင်ပို့လုပ်ရမှာ ဖြစ်တယ်။

လိုက်ဘရီ

လိုက်ဘရီ (*library*) ဆိုတာ ပညာရပ်နယ်ပယ် တစ်ခုအတွက် ရည်ရွယ်ရေးထားတဲ့ ပရိုဂရမ်ကုဒ်တွေပါပဲ။ သချုပ်အတွက်အချက် လိုက်ဘရီ ဂိမ်းရေးဖို့ လိုက်ဘရီ၏ 2D/3D ဂရပ်ဖစ်ဆဲဖို့ လိုက်ဘရီ၊ အောအိုင်အတွက် လိုက်ဘရီ စသည်ဖြင့် နယ်ပယ်အသီးသီး၊ ကိစ္စရဲပ်အဖို့ဖို့အတွက် သက်ဆိုင်ရာ ကျွမ်းကျင်ပညာရှင်တွေ ထုတ်လုပ်ဖြန့်ချီပေးထားတဲ့ လိုက်ဘရီတွေရှိတယ်။ Matrix အော်ပရေးရှင်းတွေအတွက် numpy၊ ဂရပ်ဖွဲ့မယ်ဆိုရင် matplotlib စတဲ့ လိုက်ဘရီတွေကို အင်ပို့လုပ် အသီးပြနိုင်ပါတယ်။ မေ့ထရစ် A ကို B နဲ့ မြောက်ရင် ဒီလိုပါ

```
from numpy import *
```

```
A = array([[1, 1, 2, 2],
           [2, 2, 1, 1],
           [2, 2, 1, 1]])
B = array([[2, 2],
           [2, 2],
           [1, 1],
           [2, 2]])
```

```
result = matmul(A, B)
```

```
print(result)
```

ရလဒ် အခုလိုထွေက်ပါမယ်။

```
[[10 10]
 [11 11]
 [11 11]]
```

ဒါကတေသာ ဘားချုပ် အတွက် numpy နဲ့ matplotlib လိုက်ဘရီ သုံးထားတာပါ။

```
from matplotlib.pyplot import *
from numpy import *
```

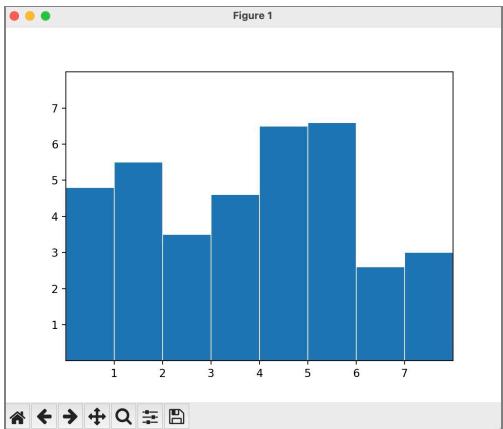
```
# make data:
x = 0.5 + arange(8)
y = [4.8, 5.5, 3.5, 4.6, 6.5, 6.6, 2.6, 3.0]
```

```
# plot
fig, ax = subplots()
bar(x, y, width=1, edgecolor="white", linewidth=0.7)

ax.set(xlim=(0, 8), xticks=arange(1, 8),
       ylim=(0, 8), yticks=arange(1, 8))

show()
```

ဘားချုတ်ကို ဒီလို ထုတ်ပေးပါတယ်။



ထဲ ၁၂

လိုက်ဘရီတွေဟာ ပရိုဂရမ်တွေ တည်ဆောက်ရာမှာ အင်မတန်မှ အရေးပါတယ်။ ဖော်ပြထားတဲ့ မေ့သရစ် နဲ့ ဘားချုတ် ကုဒ်တွေကို (အခုတော့) နားလည်မှာ မဟုတ်သေးဘူးပေါ့။ ဒါပေမဲ့ သက်ဆိုင်ရာ လိုက်ဘရီတွေနဲ့ ဒီလိုကိစ္စတွေကို သိပ်မခက်ခဲဘဲ လုပ်လိုရနိုင်တယ်ဆိုတာ မြင်မယ် ထင်ပါတယ်။ လိုက်ဘရီတွေ သာမရှိရင် ပရိုဂရမ်တွေကို အခုထက် အဆပေါင်းများစွာ အချိန်ပေးပြီး ရှုပ်ရှုပ်ထွေးထွေး ခက်ခက်ခဲ့လဲ တည်ဆောက်ကြရမှာပါ။

တံက်၊ ဓမ္မတံက်နှင့် ဆင်းတက်ခံ

ဂျိန်ပန်စာ၊ ပြင်သစ်စ စတဲ့ လူဘာသာစကားတွေဟာ စကားလုံးတွေ ဝါကျတွေနဲ့ ဖွဲ့စည်းထားသလို ပရီ ဂရမ်ကုဒ်တွေဟာလည်း စကားလုံးတွေ၊ ဝါကျတွေနဲ့ ဖွဲ့စည်းထားတာပါပဲ။ Programming language တွေမှာ စကားလုံးတွေကို တံက် (token) လိုက်ပြီး ဝါကျတွေကိုတော့ စတိတ်မန် (statement) လို ခေါ်ပါတယ်။ ဝါကျတွေကို စကားလုံးတွေနဲ့ ဖွဲ့စည်းထားသလို စတိတ်မန်တွေကတော့ တံက်တွေနဲ့ ဖွဲ့စည်းထားတာပါ။ စတိတ်မန် ပုံစံတစ်မျိုးကို တွေ့ခဲ့ပြီးပါပြီ။ အဲဒါကတော့ ရေ့စာမျက်နှာက အင်ပိုစတိတ်မန်ပဲ ဖြစ်ပါတယ်။

လူဘာသာစကားတွေမှာ ဖွဲ့စည်းတည်ဆောက်ပုံ စထရက်ချာရှိသလို programming language တွေမှာလည်း စထရက်ချာရှိဖို့ လိုအပ်တာပေါ့။ ဖွဲ့စည်းပုံ စထရက်ချာ မှန်/မမှန်ကို သဒ္ဓါစည်းမျဉ်တွေနဲ့ ထိန်းကွာ်ထားတာပါ။ ပရိုဂရမ်ကုဒ် စထရက်ချာ မှန်/မမှန် ထိန်းကွာ်ပေးတဲ့ သဒ္ဓါစည်းမျဉ်တွေကိုတော့ ဆင်းတက်စ် (syntax) လို ခေါ်ပါတယ်။

မြန်မာလိုရေးရင် မြန်မာသဒ္ဓါကို လိုက်နာရသလို Python နဲ့ ရေးရင် Python ဆင်းတက်စ်ကို

လိုက်နာရမှာပေါ့။ မြန်မာသွေ့မှားရင် ဖတ်တဲ့သူက သည်းခံနားလည် ပေးပေမဲ့ ဆင်းတက်စုံမှားရင်တော့ Python က လုံးဝ လက်ခံမှာ မဟုတ်ပါဘူး။ ဆင်းတက်စုံ စဉ်းကမ်းတွေဟာ ပိုပြီး တင်းကျပ်တယ်။ လွှဲချော်လို့ မရဘူး။ ဆင်းတက်စုံမှားနေတဲ့ ပရိုဂရမ်ကို Python က run ခွင့် ပြုမှာမဟုတ်ဘဲ အမှားနဲ့ သက်ဆိုင်တဲ့ အယ်ရာမက်ဆွဲချုပ်တွေ ပြပေးမှာပါ။ ဖြစ်လေရှိတဲ့ ဆင်းတက်စုံအမှားတွေကို မကြာခင် တွေ ရပါမယ်။

Keywords

`from, import, def, if` စတာတွေဟာ keyword တွေဖြစ်တယ်။ Python ရေးတဲ့အခါ သူနေရရာနဲ့ သူ အထိပိုယ်ကိုယ်စိန့် အသုံးပြုရတဲ့ စကားလုံးတွေဖြစ်တယ်။ `from` နဲ့ `import` ကို လိုက်ဘရီ အင်ပိုလုပ် ဖို့ သုံးတယ်။ `def` ကို ဖန်ရှင် သတ်မှတ်တဲ့အခါ သုံးတယ်။ Python က သတ်မှတ်ထားတဲ့ နေရာတွေက လွှဲလို့ အခြားကိစ္စတေအတွက် keyword တွေကို အသုံးပြုလို့ မရပါဘူး။ ဒါကြောင့် keyword တွေကို reserved word လို့လည်း ခေါ်တယ်။

main ဖန်ရှင်

‘Meet Karel’ ပရိုဂရမ် အင်ပိုစတိတ်မန့် အပြီးမှာ တွေ့ရတာကတော့ `main` ဖန်ရှင်သတ်မှတ်ချက်ပါ။ ကြည့်ရအဆင်ပြေအောင် သူချည်းသီးသန့် တစ်ဖြတ် ပြန်ပြပေးထားပါတယ်။

```
def main():
    """Karel code goes here!"""
    move()
    move()
    move()
    pick_beeper()
    turn_left()
    move()
    move()

    turn_left()
    turn_left()
    turn_left()

    move()
    put_beeper()
# End of main
```

ဖန်ရှင် (*function*) ဆိုတာ ကိစ္စတစ်ခု လုပ်ဆောင်ပေးဖို့အတွက် ယူနစ်တစ်ခုအနေနဲ့ ဖွဲ့စည်းထားတဲ့ ပရိုဂရမ်ကုဒ် အစုအဝေးတစ်ခုပါပဲ။ ဖန်ရှင်ကို အသုံးပြုတဲ့အခါ ငင်းရဲ့ လုပ်ငန်းတာဝန်အတိုင်း ဖန်ရှင် က လုပ်ဆောင်ပေးမှာ ဖြစ်တယ်။ ဖန်ရှင်အသုံးပြုတာကို ‘ဖန်ရှင်ကောလ်’ (*function call*) လုပ်တာ လို့ ပြောတယ်။

`main` ဖန်ရှင်သတ်မှတ်ချက်ကို အပိုင်းနှစ်ပိုင်းခဲ့ ကြည့်နိုင်တယ်။ ပထမတစ်ပိုင်း

```
def main():
```

ကို ဖန်ရှင်ခေါင်းစီး (function header) လို့ခေါ်တယ်။ ဖန်ရှင်ခေါင်းစီးမှာ ဖန်ရှင်နံမည်နဲ့ ဖန်ရှင်ပါရာ မိတာတွေကို ပိုက်ကွင်းထဲမှာ သတ်မှတ်ပေးရပြီး colon ‘:’ နဲ့ အဆုံးသတ်တယ်။ ဥပမာ x, y ပါရာ

မီတာ နဲ့ myfun ဖန်ရှင် အတွက်

| **def myfun(x, y):**

ပါရာမီတာမပါရင်လည်း ပိုက်ကွင်းအလဲတဲ့ တစ်စုံ ‘()’ တော့ပါရမယ်။ main ဖန်ရှင်မှာ ပါရာမီတာ မပါဘူး။ ပါရာမီတာတွေအကြောင်း နောက်ပိုင်းအခန်းတွေမှာ အသေးစိတ် လေ့လာရမှာပါ။ ကားရဲ့လုပ်ပရိုကရမ် တွေမှာ ပါရာမီတာအကြောင်း သိမ့်မလိုသေးပါဘူး။ ပါရာမီတာ မလိုတဲ့ ဖန်ရှင်တွေပဲ တွေ့ရမှာပါ။

ဖန်ရှင်ခေါင်းစဉ်းအောက် ဒုတိယပိုင်းကတော့ main ဖန်ရှင် ကုဒ်ဘလောက် (code block) ပါ။ ကုဒ်ဘလောက် ဆိုတာ ကုဒ်တွေကို အုပ်စုတစ်စုံ အဖြစ် ဖွဲ့စည်းထားတာကို ဆိုလိုတာပါ။ ဘလောက်လို့ လည်း ခေါ်တယ်။ ဘလောက်တစ်ခုမှာ ပါဝင်တဲ့ ကုဒ်လိုင်းတွေကို ညာဘက် ခွာရေးရပါမယ်။ အင်ဒန်ထုတ် (indent) လုပ်တာလို့ ခေါ်တယ်။ တက်သာကို တစ်ခုချက် (သို့) စပော်လေးခုစာ ခွာလေ့ရှိတယ်။ ဘော်ဒီပထမတစ်ဦးကြောင်း

| *"""Karel code goes here!"""*

ကို docstring လို့ ခေါ်တယ်။ `quotation` သုံးခုတဲ့ “”” တစ်စုံကြား ညာပြရေးတဲ့ ကွန်းမန်းတစ်မျိုးလို့ ယူဆ နိုင်တယ်။ ဖန်ရှင်နဲ့ ပါတ်သက်တဲ့ ရှင်းလင်းဖော်ပြချက်တွေ ရေးဖို့အတွက် သုံးတာပါ။

Docstring အောက်မှာ တွေ့ရတာကတော့ ကားရဲ့လုပ်ကွန်းမန်းတွေဆိုတာ သိပါလိမ့်မယ်။ ကားရဲ့လုပ်ကွန်းမန်းတွေဟာ stanfordkarel လိုက်ဘရဲ့ အားဖြင့် stanfordkarel လိုက်ဘရဲ့မှာ ကားရဲ့လုပ်ကွန်းမန်းတွေအတွက် ဖန်ရှင်သတ်မှတ်ချက်တွေ ပါဝင်တယ်။ ဖန်ရှင်တစ်ခုကို အသုံးပြုဖို့အတွက် အဲဒီဖန်ရှင်ကို ခေါ်ခြုံပါတယ်။ ဒါကို ဖန်ရှင်ကောလ် (function call) လုပ်တယ်လို့ ပြောတယ်။ ကားရဲ့လုပ်ကို ဘယ်ဘက်လှည့်စေချင်ရင် turn_left ဖန်ရှင်ကောလ် လုပ်ရပါမယ်။ ဘိပါကောက်ခိုင်းချင်ရင် pick_beeper ဖန်ရှင်ကောလ် လုပ်ရပါမယ်။ ဖန်ရှင်ကောလ် လုပ်တဲ့ ပုံစံက

`turn_left()`
`pick_beeper()`

စသည်ဖြင့် ဖြစ်တယ်။

Python မှာ အင်ဒန်ထုတ်ကို ဖြစ်ကတတ်ဆန်း လုပ်လို့မရဘူး။ ဘေးမျဉ်းကနေ ခွာတဲ့ အကွာအဝေး မညီတာနဲ့ ဆင်းတက်စုံအမှား ဖြစ်တယ်။ မလိုတဲ့ နေရာမှာလည်း ခွာရေးလို့ မရဘူး။ ခေါ်းစီးကို ဘေးမျဉ်းနဲ့ ခွာထားကြည့်ပါ။ အယ်ရာဖြစ်တာကို တွေ့ရမယ်။ အင်ပိုစတိတ်မှန်းလည်း ဘေးမျဉ်းနဲ့ ကွာနေလို့မရဘူး။ အခြား language တွေမှားလည်း အင်ဒန်ထုတ်လုပ် ရေးကြပေမဲ့ Python မှာလောက် မတင်းကျပ်ဘူး။ အင်ဒန်ထုတ်မလုပ်လည်း ဆင်းတက်စုံမှားတာ မဖြစ်ဘူး။

ကားရဲ့လုပ်ကွန်းမန်းတွေဟာ main ဟာ အထူးတာဝန်တစ်ခု လုပ်ဆောင်ပေးရတယ်။ အဲဒီကတော့ ပရီဂရမ်င်းဒီးမှာ **Run Program** ခလုတ် (ပုံ ၁၅ မှာကြည့်ပါ) နှိပ်လိုက်ရင် တုံ့ပြန် လုပ်ဆောင်ပေးရတာပါ။ ဒါကြောင့် ကွန်းမန်းတွေဟာ အဲဒီခလုတ် နှိပ်တော့မှပဲ စအလုပ်လုပ်တာ ဖြစ်တယ်။

Entry Point

‘Meet Karel’ ပရိုကရမ်းမှာ main ဖန်ရှင်နောက် အောက်ဆုံးအပိုင်းဟာ ပရိုကရမ် run တဲ့အခါ ပထမဆုံး စတင်လုပ်ဆောင်ပေးရမဲ့ ဖန်ရှင်ကို ဖော်ပြပေးတာပါ။ အန်ထရီပိုင် (entry point) လို့ခေါ်တယ်။

| `if __name__ == "__main__":`
| `run_karel_program("meet_karel")`

`run_karel_program` ဖန်ရှင်ဟာ ကားခဲ့ပရိုကရမ တစ်ခုအတွက် အန်ထရီပိုင့် ဖြစ်တယ်။ ပရိုကရမ တက်လာတာနဲ့ တစ်ပါတည်း ခေါ်တင်ချင်တဲ့ ကဗ္ဗာကို ဒီဖန်ရှင်မှာ ထည့်ပေးပါတယ်။ `meet_karel.w` ကဗ္ဗာကို စ စချင်းခေါ်တင်ထားချင်ရင် "meet_karel" ထည့်ပေးရမယ်။ ကဗ္ဗာဖိုင်မရှိရင် အယ်ရာတက်ပြီး ပရိုကရမ ပွင့်လာမှာ မဟုတ်ဘူး။ ကဗ္ဗာမထည့်ပေးထားဘဲ ဒီလို

```
| run_karel_program()
```

ဆိုရင် 8×8 အချယ် default ကဗ္ဗာကို တင်ပေးပါတယ်။

ကားခဲ့လ်ကဗ္ဗာတစ်ခုကို လိုချင်တဲ့ပုံစံ ဒီဇိုင်းဆွဲပြီး ဖိုင်နဲ့ သိမ်းထားရတာပါ။ ကဗ္ဗာ ပုံစံချုတဲ့ ပရီ ဂရမ်လည်း ရှိတယ်။ ကဗ္ဗာဖိုင်တွေက .w ဖိုင် အိပ်စာန်းရှင်းနဲ့ ဖြစ်တယ်။ ဒီစာအုပ်မှာပါတဲ့ ဥပမာတွေ လေ့ကျင့်ခန်းတွေ အားလုံးအတက် လိုအပ်တဲ့ ကဗ္ဗာတွေကို အဆင်သင့်ပေးထားမှာပါ။ ကိုယ့်ဟာကို လုပ်ဖို့ မလိုဘူး။ စိတ်ဝင်စားရင် စမ်းကြည့်လိုရအောင် စာမျက်နှာ (၅၃) နောက်ဆက်တဲ့ (ခ) မှာ အကျဉ်းဖော်ပြုပေးထားပါတယ်။

၁.၃ ကားခဲ့လ် ပရိုကရမ run ခြင်း

လိုအပ်တဲ့ဆော့ဖို့တွေ ထည့်သွင်းနည်းကို စာမျက်နှာ (၂၉) နောက်ဆက်တဲ့ (က) မှာ တစ်ဆင့်ချင်း ဖော်ပြပေးထားပါတယ်။ အခုက Python ပရိုကရမတစ်ခုကို အရိုးရှင်းဆုံး (လွယ်တယ်လို့ မဆိုလို) run လို့ရတဲ့ နည်းကိုဖော်ပြပေးမှာပါ။ သဘောတရားပိုင်း နားလည်ဖို့ အထောက်အပံ့ဖြစ်မယ်။ အခုနည်းလမ်းကို အကြမ်းဖျဉ်း နားလည်အောင် ဖတ်ပြီးမှ နောက်ဆက်တဲ့ (က) ကို ဖတ်စေချင်ပါတယ်။

မိုက်ခရီးဆော့ဖို့ ဝင်းပီးမှာ Notepad ၊ အက်ပလ် မက်ခံအိုအက်စ်မှာTextEdit စတဲ့ တက်စ် အယ်ဒီတာတစ်ခုခုနဲ့ ပရိုကရမက်းရေးလိုရတယ်။ ကုဒ်ဖိုင်ကို .py အိပ်စာန်းရှင်းနဲ့ သိမ်းရပါမယ်။ ပလိုန်းတက်စ် (plain text) ဖိုင် ပါပဲ။ Python ကုဒ်ဖိုင်ထို့ .txt အစား .py နဲ့ သိမ်းတာပါ။ Python ဖိုင်နံမည်ကို စာလုံးအသေးနဲ့ပဲ ပေးတဲ့ ထုံးစံရှိတယ်။ စပော်နေရာမှာ _ (underscore) သုံးတဲ့ ထုံးစံရှိတယ်။ ဒါကြောင့် 'Meet Karel' ပရိုကရမက်းကို `meet_karel.py` ဖိုင်မှာ သိမ်းသင့်တယ်။

Python နဲ့ ရေးထားတဲ့ ပရိုကရမကို run မယဆိုရင် Python ဆော့ဖို့ရှိရမှာပါ။ ဒီဆော့ဖို့ အင်စတောလ် လုပ်နည်းကို နောက်ဆက်တဲ့ (က) စာမျက်နှာ (၃၉) မှာ ဖော်ပြပေးထားပါတယ်။ Python ကုဒ်တွေကို ကွန်ပျူးတာက တိုက်ရှိက် နားမလည်ပါဘူး။ Python ဆော့ဖို့ပဲဟာ Python ကုဒ်တွေကို တိုက်ရှိက် နားလည်ပြီး ကွန်ပျူးတာပေါ်မှာ run လိုရအောင် ကြားခံဆောင်ရွက်ပေးတဲ့ ဆော့ဖို့လို ယေဘုယျအားဖြင့် ယူဆနိုင်တယ်။

Python ထည့်ပြီးရင် `stanfordkarel` လိုက်ဘရီကို အောက်ပါကွန်မန်းနဲ့ အင်စတောလ် လုပ်ရပါမယ်။ အင်တာနက်ပေါကနေ ဒေါင်းလှုပ် လုပ်ရတာမျိုးလို ကွန်နက်ရှင်ရှိရမယ်။

```
pip install stanfordkarel
```

ကားခဲ့လ်ပရိုကရမမှာ ခေါ်တင်ချင်တဲ့ ကဗ္ဗာဖိုင်တွေလည်း ရှိရပါမယ်။ `meet_karel.w` ကဗ္ဗာဖိုင်က `meet_karel.zip` ဖိုင် `worlds` ဖို့ဒါထဲမှာ ရှိပါတယ်။ <http://tinyurl.com/3mmmm9c7j> လင့်ကနေ `meet_karel.zip` ဖိုင်ကို ဒေါင်းလှုပ်လှုပ်ပါ။ ဒါ zip ဖိုင်ထဲက `worlds` ဖို့ဒါကို `meet_karel.py` ဖိုင်ရှိတဲ့ နေရာမှာ ကော်ပိုကုးထည့်ပါ။ ဝင်းပီးမှာ Command Prompt ၊ မက်ခံအိုအက်စ်မှာ Terminal ဖွင့်ပြီး `cd` ကွန်မန်းနဲ့ ကုဒ်ဖိုင်ထားတဲ့ ဖို့ဒါထဲကို သွားပြီး အောက်ပါအတိုင်း `python` ကွန်မန်းနဲ့ ကုဒ်ဖိုင်ကို run ပေးရပါမယ် (လာမ့်စာမျက်နှာမှာ နှမူနာပြထားတာ ကြည့်ပါ)။

```
python meet_karel.py
```

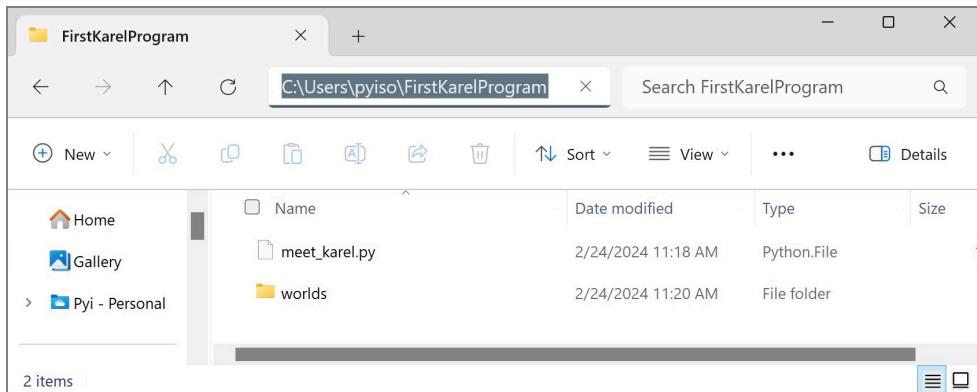
ကုဒ်ရေးထားတာ ဆင်းတက်စ်အမှား မရှိဘူးဆုံးရင် ကားခဲ့ပရိုကရမ ပွင့်လာမှာပါ။

အခုဖော်ပြခဲ့တာက ကားရဲလ်ပရိုဂရမ်တစ်ခု run ဖို့ မဖြစ်မနေလုပ်ရမဲ့ အဆင့်ဆုံးလိုအပ်ချက်ပါ။ Python ဆော့ဖို့ ရှိရမယ်။ stanfordkarel လိုက်ဘရဲ အင်စတောလ် လုပ်ရမယ်။ ကမ္ဘာဖိုင်ပါတဲ့ worlds ဖို့ရှိရမယ်။ .py ဖိုင် တစ်ခုနဲ့ ပရိုဂရမ်ကုဒ်ကို သိမ်းရမယ်။ worlds ဖို့ခိုနဲ့ ကုဒ်ဖိုင်ကို တစ်နေရာ တည်းမှု ထားရမယ်။ ပြီးရင် ကုန်မန်းလိုင်းမှာ

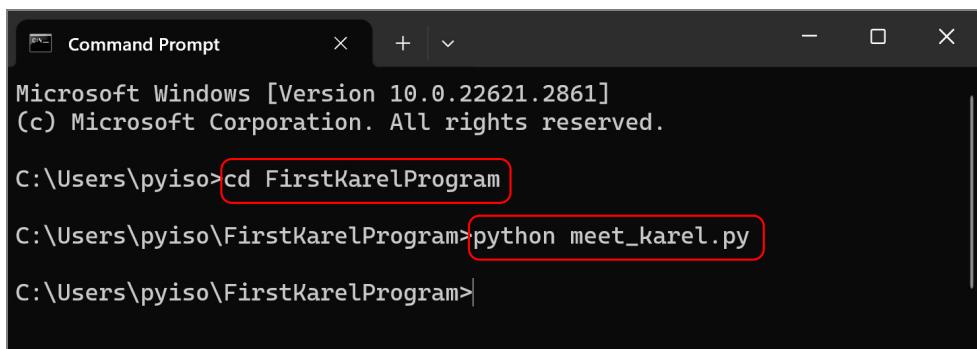
```
python your_karel_program.py
```

run የሚበ

C:\Users\pyiso\FirstKarelProgram ဖို့ဒါထဲမှာ ကုဒ်ဖိုင်နဲ့ worlds ဖို့ကို ထားပြီး ဘယ်လို run ရလဲ နမူနာပြထားတာကို ကြည့်ပါ။



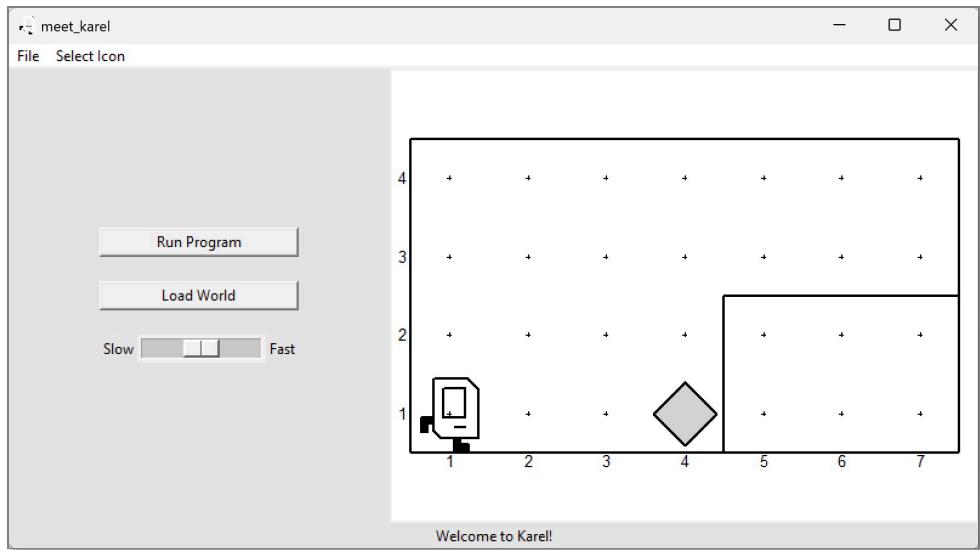
०.१



०.६

0.9 Move Beeper to Other Side

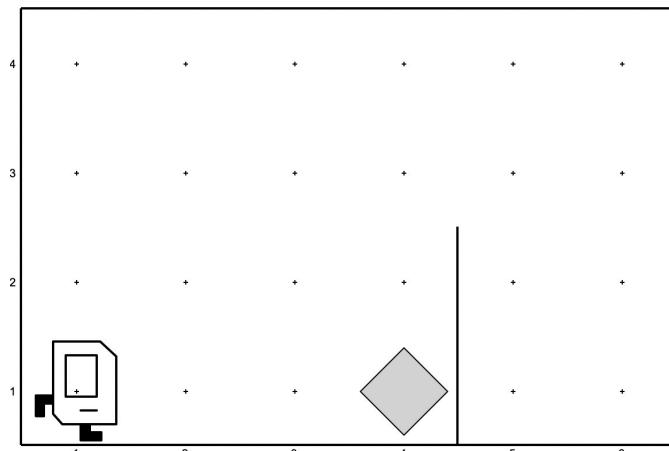
ပရိုဂေမ်းမင်း လေ့လာတဲ့အခါ တာချည်းပဲ ဖတ်နေပြီး အမှန်တကယ် နားလည်သွားဖို့ဆိုတာ မဖြစ်နိုင်ပါဘူး။ လက်တွေ့ စမ်းသပ်ကြည့်၊ ရေးကြည့်မှာ တကယ် နားလည်လာမယ်။ တကယ်လည်း ကျွမ်းကျွမ်းကျင်ကျင် ရေးတတ်တော်မယ်။ ဒါခြကြာ့င် လက်တွေ့ရေးကြည့်ပါ။ များများ လေ့ကျင့်ပါ။ ဥပမာဏ္ဍာကိုလည်း နားလည် အောင် ဖတ်ပြီးရင် မိမိဘာသာ အလတ် ပန်ရေးကြည့်ပါ။



ပုံ ၁.၅

worlds ဖို့အမှာပဲ အခု ကမ္ဘာစိုင် ထည့်ပေးထားပါတယ်။ move_beeper_to_other_side.w နံမည့်ပါ။ အန်ထရိပိုင့်အတွက် အခုလိုရေးရပါမယ်။

```
if __name__ == "__main__":
    run_karel_program("move_beeper_to_other_side")
```



ပုံ ၁.၆

အခန်း J

ကားရဲလိန့် ကွန်ထရီးလ် စတိတ်မန်များ

ကားရဲလ်ပရိုကရမ်တစ်ခု ဖွဲ့စည်းတည်ဆောက်ပုံကို ရှုံးအခန်းမှာ လေ့လာခဲ့ကြပြီး ကွန်ထရီးလ် စတိတ်မန် တွေ ဖြစ်ကြတဲ့ **for** loop, while loop, **if** နဲ့ **if...else** တို့ကို အခုဆက်လက် လေ့လာကြပါမယ်။ ပရိုကရမ်တစ်ခုကို run တဲ့အခါ ပရိုကရမ်ကြိုင်ထဲက ညွှန်ကြားချက်တွေအတိုင်း ကွန်ပျူးတာက လုပ်ဆောင် ပေးတာပါ။ ဒီလိုလုပ်ဆောင်ပေးတာကို အကွွဲစီကျိုး (execute) လုပ်တယ်လို့ ခေါ်တယ်။ ကွန်ထရီးလ် စတိတ်မန်တွေဟာ ပရိုကရမ်တစ်ခုရဲ့ execution flow ကို ထိန်းချုပ်ပေးတာ ဖြစ်တဲ့အတွက် control flow statements တွေလိုလည်း ခေါ်ပါတယ်။

J.၁ for loop

ကွန်ထရီးလ်စတိတ်မန် တစ်မျိုးဖြစ်တဲ့ **for** loop ဟာ စတိတ်မန် တစ်ခု (သို့) စတိတ်မန် တစ်စုကို သတ်မှတ်ထားတဲ့ အကြိမ်အရေအတွက် ပြည့်အောင် ထပ်ခါထပ်ခါ ပြန်ကျော့ပေးပါတယ်။ move ကို နှစ် ဆယ့်ငါးကြိမ် ကျော့ပေးဖို့ for loop နဲ့ အခုလို

```
for i in range(25):
    move()
```

ရေးရပါတယ်။ put_beeper, move, turn_left စတိတ်မန် သုံးကြာင်းတစ်စုကို အကြိမ်တစ်ရာ ကျော့ချင်ရင် ဒီလိုပါ

```
for i in range(100):
    put_beeper()
    move()
    turn_left()
```

နှစ်ဆယ့်ငါးကြိမ်ကို range(25), အကြိမ်တစ်ရာကို range(100) စသည်ဖြင့် တွေ့ရတယ်။ ယော ဘုယျအားဖြင့် အကြိမ်အရေအတွက် N ကြိမ် ကျော့ချင်ရင်

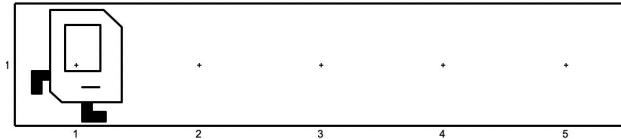
```
for x in range(N):
    statement1
    statement2
    statement3 etc.
```

ပုံစံနဲ့ သတ်မှတ်ရတာ။ N က အကြိမ်အရေအတွက်ကို ဖော်ပြတဲ့ ကိုန်းပြည့်ကြောန်းဖြစ်တယ်။ for loop

ရေးတဲ့အခါ ကော်လံ : မကျန်ခဲ့ဖို့ သတိပြုရပါမယ်။ x ဟာ ပေါ်ရောကဲလ်တစ်ခုဖြစ်ပြီး i , j , k စတဲ့ အကွ္ခာတစ်ခုနဲ့ ကိုယ်စားပြုလေ့ရှိတယ်။

ပြန်ကျော့စေချင်တဲ့ စတိတ်မန်တွေကို for ရဲ့ ညာဘက်ကို အင်ဒန်ထု လုပ်ပေးရပါမယ်။ အောက်ပါ အတိုင်းဆိုရင် put_beeper ကိုပဲ အကြိမ်တစ်ရာ လုပ်မှာပါ။ move နဲ့ turn_left ပြန်ကျော့မဲ့ထဲမှာ မပါတော့ဘူး။

```
for i in range(100):
    put_beeper()
move()
turn_left()
```



ပုံ J.၁

ပုံ (J.၁) ကားရဲလ်ကဲ့မှာ ကွန်နာအားလုံးမှာ ဘိပါတစ်ခုစီ ချထားပေးရမယ်။ ကွန်နာဝါးခုရှိတာမို့ လို့ စုံပေါင်း ဘိပါဝါးခဲ့ ချထားရမှာပါ။ move ကတော့ လေးကြိမ်ပဲ လုပ်ရမယ် (ကားရဲလ်ရှေ့မှာ ကွန်နာ လေးခုပဲ ရှိတယ်)။ ဒီအတွက်ကို အခုလိုရေးဆိုင်ပါတယ်။

```
# File: make_row_of_5_beeper.py
from stanfordkarel import *

def main():
    for i in range(4):
        put_beeper()
        move()

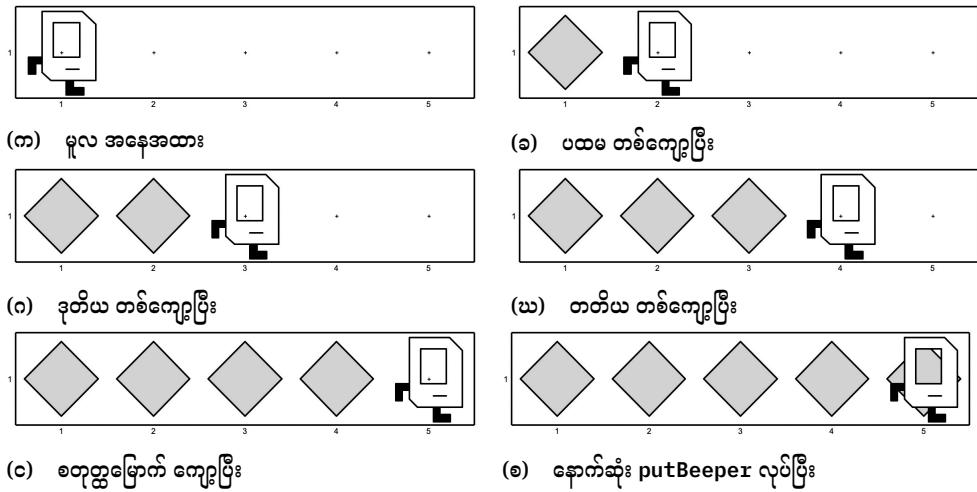
    put_beeper()

if __name__ == "__main__":
    run_karel_program("make_row_of_5_beeper")
```

for loop ဟာ put_beeper နဲ့ move ကို လေးကြိမ်ကျော့ပေးမှာပါ။ တစ်ကြိမ်ပြီးတိုင်း ရှိနေမဲ့ အနေအထား တစ်ခုချင်းစိတိ ပုံ (J.၂) မှာ တွေ့ရှိနိုင်ပါတယ်။ လေးကြိမ်မြောက် နောက်ဆုံးတစ်ကျော့အပြီး ပုံ (J.၂) (c) မှာ ဘိပါတစ်ခုလုံးနေသေးတာ တွေ့ရမယ်။ for loop ပြီးတဲ့အခါ put_beeper တစ်ခါ ထပ်လုပ်ပေးတော့မှ တစ်တန်းလုံးအပြည့်ဖြစ်သွားမှာပါ။

Off-by-one bug

ပြီးခဲ့တဲ့ဥပမာမှာ for loop အပြီး put_beeper မလုပ်မိရင် ပရှိကရမ်ဟာ လိုချင်တဲ့အတိုင်း မဖြစ်ပါဘူး။ ဘိပါတစ်ခုလုံးနေမယ်။ ဒီလိုပြဿနာမျိုးဟာ အဖြစ်များတဲ့ အမှားဖြစ်ပြီး off-by-one bug လို့ခေါ်ပါတယ်။ Loop သုံးတဲ့အခါ ကြိုရလေ့ရှိတဲ့ ဖြစ်တတ်တဲ့ အမှား (bug) ဖြစ်တယ်။ သတ်မှတ်ထားတဲ့အတိုင်း၊ ဖြစ်



ဦး J-J

သင့်တဲ့အတိုင်း ပရိုဂရမ်က အလုပ်မလုပ်ဘဲ ဖြစ်နေတဲ့ အမှားကို ကွန်ပျူးတာအသုံးအနှစ်နဲ့မှာ ပျော်လိုက်ခြင်း၏ဘက်ပြင်ပေးတာကိုတော့ debug လုပ်တယ်လို့ခေါ်ပါတယ်။

ဟိပါတစ်ခုကျွန်ခဲ့တဲ့ အမှားမျိုးဟာ off-by-one bug သာဓကတစ်ခုမျှသာ ဖြစ်တယ်။ ခုနှစ်နဲ့ ဆယ့်သုံးကြား ကဏ္န်း ခုနှစ်လုံးရှိပါတယ် (ခုနှစ်နဲ့ ဆယ့်သုံးအပါဝင် ဆိုလျှင်)။ $13 - 7 = 6$ လိုတွေ့က်မိရင် တစ်လုံး လိုနေပါလိမ့်မယ်။ သုံးပေါ်းတစ်တိုင် အေလျားပေသုံးဆယ်ရှိ တပြောင့်တည်း ခြေစည်းရုံးတစ်ခု အတွက် တိုင်စုစုပေါင်း တစ်ဆယ့်တစ်တိုင် လိုပါမယ်။ $30 \div 3 = 10$ လို့ တွေ့က်ရင် မမှန်ပါဘူး။ တစ်ခု လိုတာ အပြင် တစ်ခုပုံးနေတာလည်း ဖြစ်တတ်တယ်။ စောနက ခြေစည်းရုံးမှာပဲ ထရံအချုပ်အရေအတွက် ကတော့ ကိုးချပ်ဖြစ်ရမှာပါ။ $30 \div 3 = 10$ လို့ တွေ့က်ရင် တစ်ခုပုံးနေပါမယ်။ ခုနှစ်နဲ့ ဆယ့်သုံးကို ထည့်မစဉ်းစားရင် ကြားမှာ ကဏ္န်းပါးလုံးရှိတာပါ။ $13 - 7 = 6$ လိုတွေ့က်မိရင် တစ်ငုံး ပို့နေပါလိမ့်မယ်။ ဒါ ဥပမာ အားလုံးဟာ အခြေခံသဘောတရားအားဖြင့် သိပ်မကွာခြားတဲ့ off-by-one bug ပုံစံကွဲအမျိုးမျိုး ဖြစ်တယ်။

‘Make Row of Five Beepers’ ခုတိယ ဗုဒ္ဓရှင်း

ပရိုဂရမ်ရေးတဲ့အခါ ပရိုဂရမ်မာတွေ တစ်ယောက်နဲ့တစ်ယောက် စဉ်းစားပုံ စဉ်းစားနည်း၊ ဖြေရှင်းနည်းထပ်တူကျလေ့မရှိပါဘူး။ ဘို့ပါငါးခဲ့ အတန်းလိုက် ချုပေးတဲ့ ပရိုဂရမ်ကိုပဲ နောက်ဗုဒ္ဓရှင်းတစ်မျိုးနဲ့ ရေးနိုင်ပါတယ်။

```
def main():
    put_beeper()
    for i in range(4):
        move()
        put_beeper()
```

for loop မစခင် put_beeper လုပ်ထားတာ၊ move နဲ့ put_beeper အစဉ်ပြောင်းသွားတာ (ပထမဗုဒ္ဓရှင်းနဲ့ ပြောင်းပြန်ဖြစ်နေတာ) သတိထားကြည့်ပါ။

J.J အင်ဒန်ထည်လုပ်ခြင်းနှင့် ကုဒ်စာတရာ်ချာ

Python ဟာ အင်ဒန်ထည်လုပ်ခြင်းဖြင့် ပရိုဂါရမ်ကုဒ် ဖွဲ့စည်းပဲ စထရက်ချာကို ဖော်ပြတဲ့ programming language ဖြစ်ပါတယ်။ Python ကုဒ်တွေကို ဘလောက် (block) တွေနဲ့ ဖွဲ့စည်းထားတယ်လို့ ရှုမှင်နိုင်တယ်။ ဘလောက်ဆိတာ ကုဒ်တွေကို အပုံစုတစ်စု အဖြစ် ဖွဲ့စည်းထားတာကို ဆိုလိုတာပါ။

```
def main():
    put_beeper()
    for i in range(4):
        move()
        put_beeper()
    pick_beeper()
    turn_left()
```

အခု Python ကုဒ်မှာ အင်ဒန်ထည်လုပ်ထားဘဲ ဘယ်ဘက်စွဲနဲ့ ကပ်ရေးထားတဲ့ **def main():** (ဖန်ရှင်ခေါင်းစဉ်း) ဟာ အပေါ်ဆုံးအဆင့် (top level) ဖြစ်တယ်လို့ ယူဆတယ်။ ဖန်ရှင်ခေါင်းစဉ်းအောက် အင်ဒန်ထည်လုပ်ထားတဲ့ ကုဒ်လိုင်းအားလုံးဟာ **main** ဖန်ရှင် ဘလောက်ထဲမှာ အကျိုးဝင်တယ်။ **pick_beeper** အထိပါတယ်။ **put_beeper** (ပထမတစ်ခု)၊ **for** loop နဲ့ **pick_beeper** တို့ဟာ ပထမအဆင့် အင်ဒန်ထုတ်ဖြစ်တယ်။

တစ်ခါ **for** အောက်မှာ ဒုတိယတစ်ဆင့် အင်ဒန်ထည်လုပ်ထားတဲ့ ကုဒ်လိုင်းအားလုံး **for** ဘလောက် ထဲမှာ အကျိုးဝင်တယ်။ **move** နဲ့ **put_beeper** ပါဝင်တယ်။ **pick_beeper** ကတော့ ပထမအဆင့် ပြန်ဖြစ်သွားတဲ့အတွက် **for** ဘလောက်ထဲမှာမဖို့ဘူး။

အောက်ဆုံး **turn_left()** ကလည်း အင်ဒန်ထည်လုပ် မထားတဲ့အတွက် အပေါ်ဆုံးအဆင့်ပဲ။ **def main():** နဲ့ အဆင့်တူတော်ပေါ့။ **main** ဖန်ရှင်ဘလောက် အပြင်ဘက်မှာလို့ ယူဆရမယ်။

အင်ဒန်ထည်လုပ်ထားတဲ့ အဆင့်ကို ကြည့်ပြီး ဘလောက်တွေရဲ့ ဖွဲ့စည်းပဲကို ကွက်ကွက်ကွင်းကွင်း ထင်းကနဲ့ မြင်နိုင်တယ်။ အပေါက်ကုဒ်ကို ကြည့်တာနဲ့ **main** ဖန်ရှင်ထဲမှာ **for** loop ရှိတယ်။ **for** loop ထဲမှာ **move** နဲ့ **put_beeper** ပါဝင်တယ်ဆိတာ သိသာတယ်။ **pick_beeper** ဟာ **for** loop အပြင်မှာ **turn_left** ဟာ **main** အပြင်မှာ ဆိုတာကို အင်ဒန်ထည်လုပ်ထားတဲ့ အဆင့်က ဖော်ပြန်တယ်။

J.2 while loop

အခြေအနေတစ်ခုပဲ မှန်နောသူ၍ စတိတ်မန်တွေကို တစ်ကြိမ်ပြီးတစ်ကြိမ် ပြန်ကျော့ လုပ်ဆောင်စေချင်ရင် **while** loop ကို အသုံးပြုပါတယ်။ **for** နဲ့ **while** loop နှစ်ခုလုံးက ပြန်ကျော့ပေးတာ ဖြစ်ပေးတဲ့ **for** ကို အကြိမ်အရေအတွက် အတိအကျိုးတဲ့ကိစ္စမျိုးမှာ သုံးလေ့ရှိပြီး **while** ကိုတော့ ဘယ်နှစ်ကြိမ်လဲ ကြိုတွက်လို့မရဘဲ အခြေအနေ အပေါ်မှုတည်ပြီး လုပ်ဆောင်ပေးရမဲ့ အကြိမ်အရေအတွက် ကွာခြားနိုင်တဲ့ ကိစ္စမျိုးတွေမှာ သုံးလေ့ရှိတယ်။ ဥပမာတစ်ခုနဲ့ ကြည့်ရင် ပို့နားလည်ပါလိမ့်မယ်။

(၁) လမ်းပေါ်မှာ ဘိပါတစ်ခုရှိမယ်။ ဘိပါ ဘယ်ကွန်နာမှာရှိမှုမှာလဲ၊ လမ်းဘယ်လောက်အရှည်ဖြစ်မလဲ ကြိုတင်မသိဘူးလို့ ယူဆပါ။ နမူနာကဗ္ဗာ တစ်ခုကို ပုံး (J.2) မှာ တွေ့ရှိနိုင်တယ်။ ဘိပါကို သွားပြီး ကောက်ခိုင်းရပါမယ်။ အလားတူ မည်သည့်ကဗ္ဗာအတွက်မဆို ဘိပါကောက်ပေးနိုင်ရမှာ ဖြစ်တယ်။ ဘိပါရှိမဲ့ ကွန်နာကို ကြိုတင်မသိတားတဲ့အတွက် ဘယ်နှစ်ကြိမ် **move** လုပ်ခိုင်းရမှုမှာလဲ မသိဖြစ်နေတယ်။ အကြိမ်အရေအတွက် မသိတဲ့အတွက် **for** loop နဲ့ အဆင်မပြေတော့ပါဘူး။

ဘိပါမရှိသူ၍ တစ်ကြိမ်ပြီးတစ်ကြိမ် **move** လုပ်ခိုင်းမယ်ဆိုရင် နောက်ဆုံးမှာ ဘိပါရှိတဲ့ကွန်နာကို ရောက်သွားမှာ သေချာပါတယ်။ အဲဒီလို့ လုပ်ခိုင်းဖို့ရာအတွက် **while** loop နဲ့ ရေးထားတာကို အခုလုံတွေ့ရပါမယ်။



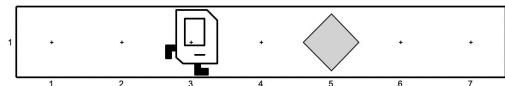
ဗိုလ်



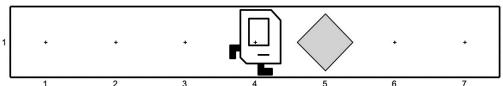
(က) မူလ အနေအထား



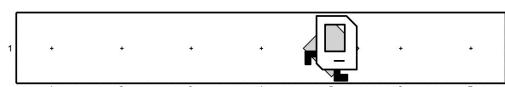
(ခ) ပထမ တစ်ကျွေးပြီး



(ဂ) ဒုတိယ တစ်ကျွေးပြီး



(ဃ) တတိယ တစ်ကျွေးပြီး



(င) စတုတွေ့ဖြောက် ကျွေးပြီး

ဗိုလ်

```
while no_beeper_present():
    move()
```

ကားရဲလ်ဟာ အခြေခံကွန်မန်း လေးခုအပြင် လက်ရှိ ကွန်နာမှာ ဘိပါရှိ/မရှိ ရှေ့မှာ နံရုပ်တ်နေ့/မနေ့ အရှေ့ဘက်ကို မျက်နှာမှုထား/မထား စတဲ့ သူနဲ့ (သို့) သူရဲကဗ္ဗာနဲ့ သက်ဆိုင်တဲ့ အခြေအနေတစ်ရုပ်ရုပ် မှန်/မှုန် စစ်ဆိုင်ပါတယ်။ no_beeper_present က လက်ရှိ သူရဲနေတဲ့ ကွန်နာမှာ ‘ဘိပါ မရှိဘူးလား’ ဆိုတဲ့ အခြေအနေ စစ်ဆိုင်းတာပါ။ လက်ရှိကွန်နာမှာ ဘိပါ ‘မရှိ’ ရင် ဒီအခြေအနေက မှန်တယ်လို့ ယူဆ ရမှာပေါ့။ အကယ်၍ ‘ရှိ’ နေရင်တော့ မှားတယ်လို့ ယူဆရမယ်။ အုမှန် (သို့) အမှား ရလဒ်ထွက်မဲ့ ဒီလို မျိုး အခြေအနေစစ် ကွန်မန်းတွေကို ကွန်ဒါရှင် (condition) လိုခြေပါတယ်။

while loop အလုပ်လုပ်ပုံက ဒီလိုပါ။ no_beeper_present ကွန်ဒါရှင် စစ်ပါတယ်။ မှန်ရင် move လုပ်ပါတယ်။ ပြီးရင် ကွန်ဒါရှင်ပြန်စစ်တယ်။ မှန်ရင် move ကို နောက်ထပ်တစ်ကြိမ် ထပ်ကျွေးပါတယ်။ ကွန်ဒါရှင်စစ်လိုက်၊ မှန်ရင် တစ်ခါထပ်ကျွေးလိုက်၊ ကွန်ဒါရှင်ပြန်စစ်လိုက်၊ မှန်ရင် နောက်တစ်ခါ ထပ်ကျွေးလိုက်၊ ... ဒီအတိုင်းဆက်ပြီး ကွန်ဒါရှင်မှန်နေသ၍ while loop က move ကို ပြန်ကျွေးပေး မှာပါ။ ကွန်ဒါရှင် စစ်လိုက်လို့ မှားသွားပြီဆုံးရှင်တော့ ထပ်မကျွေးတော့ဘဲ ရပ်သွားမှာဖြစ်တယ်။ while loop တစ်ကြိမ်ကျွေးပြီးတိုင်း ရှိနေမဲ့ အနေအထား တစ်ခုချင်းကို ပုံ (၂.၄) မှာ လေ့လာကြည့်ပါ။ လေးကြိမ်မြောက် ကျွေးပြီးနောက် ကွန်ဒါရှင်စစ်လိုက်တဲ့အခါ မှားနေပြီ။ ဒီအခါ while loop က ထပ်မကျွေးတော့ဘဲ ရပ်သွား။ ဒီလို loop က ပြန်ကျွေးနေတာ ရပ်သွားရင် loop ကနေ ထွက်တယ် (loop exits) လို့ ဖြေ လေ့ရှိတယ်။

အပြည့်အစုံ ဖော်ပြပေးထားတဲ့ ပရိုဂရမ်ကို လေ့လာကြည့်ပါ။

```
# File: go_pick_beeper.py
from stanfordkarel import *

def main():
    while no_beeper_present():
        move()

    pick_beeper()

if __name__ == "__main__":
    run_karel_program("go_pick_beeper")
```

ကားရဲလ် ကွန်ဒီရှင်များ

`beepers_present` ကွန်ဒီရှင် ကျတော့ လက်ရှိကွန်နာမှာ ဘိပါရိရင် အမှန်၊ မရှိရင် အမှား ရလဒ်ထွက်တယ်။ `no_beeper_present` နဲ့ ဆန်ကျင်ဘက်ပေါ့။ ကားရဲလ် ကွန်ဒီရှင်တွေအားလုံးကို တော်လ် (၂၃) မှာ ကြည့်ပါ။ ပုံမှန်စစ်တာနဲ့ အပြင်းပုံစစ်တာ နှစ်မျိုးကို ယူဉ်တွဲပြထားပါတယ်။

ကွန်ဒီရှင်	ဆန်ကျင်ဘက် ကွန်ဒီရှင်	စစ်ပေးသည့် အခြေအနေ
<code>front_is_clear</code>	<code>front_is_blocked</code>	ရှေ့မှာ နံရံကပ်လျက် ရှိမရှိ
<code>left_is_clear</code>	<code>left_is_blocked</code>	ဘယ်ဘက်မှာ နံရံကပ်လျက် ရှိမရှိ
<code>right_is_clear</code>	<code>right_is_blocked</code>	ညာဘက်မှာ နံရံကပ်လျက် ရှိမရှိ
<code>beepers_present</code>	<code>no_beeper_present</code>	လက်ရှိကွန်နာမှာ ဘိပါရိမရှိ
<code>beepers_in_bag</code>	<code>no_beeper_in_bag</code>	ကားရဲလ်၏ ဘိပါအိပ်ထဲ ဘိပါရှိမရှိ
<code>facing_north</code>	<code>not_facing_north</code>	အရှေ့ဘက် မျက်နှာမှူလျက် ရှိမရှိ
<code>facing_east</code>	<code>not_facing_east</code>	အနောက်ဘက် မျက်နှာမှူလျက် ရှိမရှိ
<code>facing_west</code>	<code>not_facing_west</code>	တောင်ဘက် မျက်နှာမှူလျက် ရှိမရှိ
<code>facing_south</code>	<code>not_facing_south</code>	မြောက်ဘက် မျက်နှာမှူလျက် ရှိမရှိ

တော်လ် ၂.၁ ကားရဲလ် စစ်နိုင်သည့် ကွန်ဒီရှင်များ

while loop ဆင်းတက်စိ

while loop ရေးတဲ့ ပုံစံက ယော့ယျအားဖြင့် ဒီလိုပါ။

```
while condition :
    statement1
    statement2
    statement3 etc.
```

`condition` က ကားရဲလ်ကွန်ဒီရှင် တစ်ခုဖြစ်တယ်။ ကော်လ် : မကျွန်းခဲ့အောင် ဂရစိုက်ပါ။ `condition` မှန်နေသေးသို့ ပြန်ကော့စေချင်တဲ့ စတိတ်မန်တော်ကို while အောက်မှာ အင်ဒန်ထဲလုပ်ထား ရပါမယ်။ ရှေ့မှာရှင်းနေသူ၏ move လုပ်တဲ့ while loop ကို အခုလို

```
while front_is_clear():
    move()
```

ရေးပါတယ်။

‘Make Beeper Row’ ဥပမာ

‘Make Row of Five Beepers’ ဥပမာမှာ လမ်းခွဲအလျေားဟာ မပြောင်းလဲဘူး ယူဆတာမို့လို့ for loop ကို အသုံးပြုတာ ဆိုလျော်ပါတယ်။ လမ်းခွဲအရှည်ကို ကြိုးမသိထားဘူး၊ တစ်လမ်းလုံး ဘိပါတွေ ဖြန့်ထားပေးရမယ်ဆိုရင် while နဲ့ ရေးရမှာပါ။

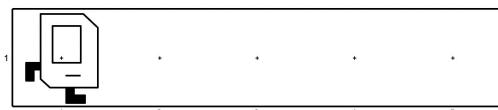
```
# File: make_beeper_row.py
from stanfordkarel import *

def main():
    while front_is_clear():
        put_beeper()
        move()

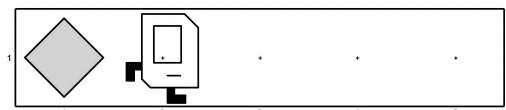
    put_beeper()

if __name__ == "__main__":
    run_karel_program()
```

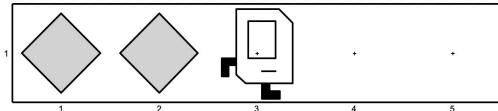
တစ်ကြိမ်ကျော်ပြီးတိုင်း ရှိနေမဲ့ အနေအထားကို ပုံ (၂၅) မှာ ကြည့်ပါ။ လေးကြိမ်မြောက်အပြီး front_is_clear စစ်တဲ့အခါ မှားနေပြီဖြစ်လို့ ထပ်မကျော်တော့ဘဲ loop ကနေ ထွက်သွားမယ်။ ဒီ အခါမှာ ဘိပါတစ်ခု လိုနေနေသေးတယ်။ put_beeper ထပ်လုပ်ရမယ်။ တစ်ခါပဲ လုပ်ရမှာပါ ဒါကြောင့် while loop ထဲမပါအောင် ပထမအဆင့် အင်ဒန်ထဲပဲ ဖြစ်ရမယ်။



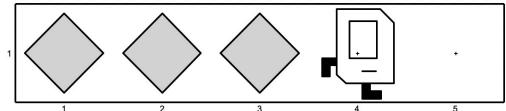
(က) မူလ အနေအထား



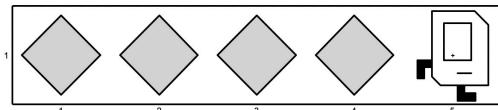
(ခ) ပထမ တစ်ကျော်ပြီး



(ဂ) ဒုတိယ တစ်ကျော်ပြီး



(ဃ) တတိယ တစ်ကျော်ပြီး



(င) စတုတွေ့ပြောက် ကျော်ပြီး

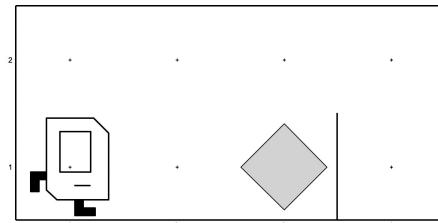
J.၄ if စတိတ်မန်

တစ်ခု (သို့) တစ်ခုထက်ပိုတဲ့ စတိတ်မန်တွေကို အခြေအနေတစ်ရပ် မှန်တော့မှပဲ လုပ်ဆောင်စေချင်တဲ့ အခါ if ကို အသုံးပြနိုင်တယ်။ ဥပမာ

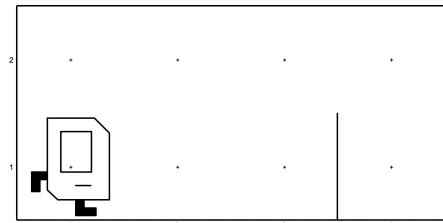
```
if beepers_present():
    pick_beeper()
```

if စတိတ်မန်ဟာ beepers_present ကွန်ဒါရှင် မှန်တော့မှပဲ pick_beeper လုပ်မှာပါ။ မှားရင် မ လုပ်ပါဘူး။

ပုံမှာတွေရှုတဲ့ ကမ္ဘာနှစ်ခုထဲက တစ်ခုမှာ ကားရဲလုံရှိနေမယ် ဆိုပါစို့။ ဘိပါရှိတဲ့ ကမ္ဘာဆိုရင် ဘိပါကို



(က)



(ခ)

ပုံ J.၆

နံရုံအခြားဘက် အောက်ခြေကို ရွှေ့ပေးရမယ်။ မရှိတဲ့ ကမ္ဘာဆိုရင် နံရုံ ဒီဘက် အောက်ခြေမှာပဲ ရပ်နေရ မယ်။ ပရိုဂရမ်က ကမ္ဘာနှစ်ခုလုံးမှာ မှန်အောင် အလုပ် လုပ်နိုင်ရပါမယ်။ ဒီအတွက် if စတိတ်မန် သုံးထားတာ လေ့လာကြည့်ပါ။

```
# File: move_beeper_to_other_side_if_any.py
from stanfordkarel import *
```

```
def main():
    move()
    move()
    if beepers_present():
        pick_beeper()
        turn_left()
        move()
        # turn_right
        turn_left()
        turn_left()
        turn_left()
        move()
        # turn_right
        turn_left()
        turn_left()
        turn_left()
        move()
        put_beeper()
        turn_left()
```

```
if __name__ == "__main__":
    run_karel_program("mbtos1")
```

if အောက်က စတိတ်မန့်တွေကို အင်ဒန်ထဲပုံတားတာ သတိပြုပါ။ အဲဒီ စတိတ်မန့်အားလုံး beepers_present ဖြစ်မှ လုပ်ဆောင်မှာ ဖြစ်တယ်။ ဘိပါမရှိတဲ့ ကမ္ဘာမှာ စမ်းကြည့်ဖို့အတွက် Load World ခလုတ်နှင်ပြီး ခေါ်တင်ပါ။ ဒါမှုမဟုတ် ပရိုဂရမ်ကုဒ်မှာ "mbtos2" လိုပြင်ပြီး ပြန် run ပါ။
if စတိတ်မန့် ယော့ယျစထရက်ချာကို အခုလုံတွေ့ရတယ်။

```
if condition :
    statement1
    statement2
    statement3 etc.
```

condition ၏ front_is_clear, beepers_present စတဲ့ ကားရဲလ် ကွန်ဒါရှင် တစ်ခုခုဖြစ်မယ်။
တေဘာ် (၂.၃) မှာ ကားရဲလ်ကို စစ်ခိုင်းလိုရတဲ့ ကွန်ဒါရှင်အားလုံး ပြထားပါတယ်။

J.၆ if...else စတိတ်မန့်

အခြေအနေတစ်ခုပဲ မှန်တဲ့အခါ လုပ်ဆောင်ရမှာနဲ့ မှုးတဲ့အခါ လုပ်ဆောင်ရမှာ မတူကဲပြားနေတဲ့အခါ if...else ကို သုံးပါတယ်။ ရှေ့က if စတိတ်မန့် ဥပမာ ပုံ (J.၆) မှာ ဘိပါမရှိရင် နိုးမှုလနေရာ ကို ပြန်လာခိုင်းချင်တယ်ဆိုပါစို့။ if...else နဲ့ အခုလုံ ရေးရပါမယ်။

```
from stanfordkarel import *
```

```
def main():
    move()
    move()
    if beepers_present():
        pick_beeper()
        turn_left()
        move()
        # turn_right
        turn_left()
        turn_left()
        turn_left()
        move()
        # turn_right
        turn_left()
        turn_left()
        turn_left()
        move()
        put_beeper()
        turn_left()
    else:
        turn_left()
        turn_left()
        move()
```

```

move()
turn_left()
turn_left()

if __name__ == "__main__":
    run_karel_program("mbtos2")

if...else စတိတ်မန့်က ကွန်ဒါရှင်မှန်ရင် if ဘလောက်ကိုလုပ်ဆောင်ပေးပြီး ကွန်ဒါရှင်မှားရင်
တော့ else ဘလောက်ကို လုပ်ဆောင်ပေးတာပါ။ ယေဘုယျပုံစံက ဒီလိုပါ

if condition :
    statement1a
    statement2a
    statement3a etc.
else:
    statement1b
    statement2b
    statement3b etc.

```

J.၆ Nested ကွန်ထရီးလ် စတိတ်မန်များ

ကွန်ထရီးလ် စတိတ်မန့် ဘလောက်ထဲမှာ ကွန်ထရီးလ် စတိတ်မန့် ရှိလိုခဲ့ပါတယ်။ Nested ကွန်ထရီးလ် စတိတ်မန့်လို့ ခေါ်ပါတယ်။

Nested for loop

ဘိပါသုံးခုချလိုက်၊ ရှေ့တိုးလိုက် လုပ်တာကို လေးကြိမ်ကျော်ချင်ရင် for loop နဲ့ အခုလို ရေးရမယ်ဆို တာ သိခဲ့ပြီးပါပြီ။

```

for i in range(4):
    put_beeper()
    put_beeper()
    put_beeper()
    move()

```

ဒါ for loop ထဲက put_beeper သုံးကြာင်းကိုလည်း နောက်ထပ် for တစ်ခုနဲ့ ရေးလို့ ရရှိပေါ့။ ဒီလို ဖြစ်သွားမှာပါ

```

for i in range(4):
    for j in range(3):
        put_beeper()
        move()

```

for loop နဲ့ခုံးဆင့်ရေးထားလို့ nested for loop လို့ခေါ်ပါတယ်။ Loop တစ်ခုချင်းကို သိုံးခြား ရည်ဗွန်းချင်တဲ့အခါ အပြင် for loop နဲ့ အတွင်း for loop လို့ ပြောလေ့ရှိတယ်။ အင်ဒ်ထဲ လုပ်ထားပုံအရ အပြင် for loop ဘလောက်ထဲမှာ အတွင်း for loop နဲ့ move ပါဝင်တယ်။ အတွင်း for loop ဘလောက်ထဲမှာတော့ put_beeper ပဲပါတယ်။ move မပါ ပါဘူး။ အပြင် for loop မှာ ဖေရီရောဘဲလ် i ဆိုရင် အတွင်းမှာ i မဖြစ်ရပါဘူး။ j, ဒါမှုမဟုတ် k မတူတာ တစ်ခုခုဖြစ်ရပါမယ်။

အတန်းလိုက် ကွန်နာ ငါးခုမှာ တစ်ကွန်နာ ဘိပါ (၂၅) ခုစီ ချထားပေးဖို့ nested loop နဲ့ ရေးထားတာကို လေ့လာကြည့်ပါ။

```
# File: row_of_beeper_piles.py
from stanfordkarel import *

def main():
    for i in range(4):
        for j in range(25):
            put_beeper()
            move()

    for i in range(25):
        put_beeper()

if __name__ == "__main__":
    run_karel_program("5x1")
```

Nested for and while

ကမ္မာပါတ်လည် ဘိပါခြိစည်းရိုး ခတ်ပေးဖို့အတွက် for နဲ့ while nest လုပ်ထားတာကို လေ့လာကြည့်ပါ။ (၁, ၁) ကွန်နာမှာ အရှေ့ဘက်လည် အနေအထားကနေ စမယ်လို ယူဆပါ။ ကမ္မာအရွယ်အစား အပိုးမျိုးကို ခြိစည်းရိုး ခတ်ပေးနိုင်ရပါမယ်။

တောင်ဘက် နံရုံတော်လျှောက် ဘိပါတွေချသွားမယ် (နောက်ဆုံး ကွန်နာမှာ ဘိပါမချသ ချန်ထားမယ်)။ ပြီးရင် အရှေ့ဘက် နံရုံအတွက် အဆင်သင့်ဖြစ်အောင် ဘယ်ဘက်လှည့်မယ်။ ပုံ (၂.၇) (က) နဲ့ (ခ) ကို ကြည့်ပါ။ အရှေ့၊ မြောက် နဲ့ အနောက်ဘက် နံရုံတွေအတွက်လည်း ဒီအတိုင်း လုပ်သွားရဲ့ပါပဲ။ ပုံ (၂.၇) (ဂ)၊ (ယ)၊ (င) အသီးသီးကို ကြည့်ပါ။

နံရုံတော်ဘက် အတွက် ဆုံးရင် အခုလို

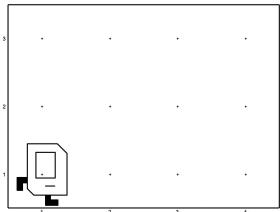
```
while front_is_clear():
    put_beeper()
    move()
    turn_left()
```

ဖြစ်မယ်။ (while ဘလောက်မှာ turn_left မပါတာ ကရုပြပါ)။ နံရုံလေးဘက်အတွက် လေးကြိမ် လုပ်ရမှာဆိုတော့ for နဲ့ အခုလို

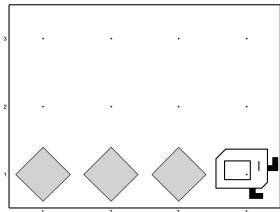
```
# File: beeper_fence.py
for i in range(4):
    while front_is_clear():
        put_beeper()
        move()
        turn_left()
```

ရေးရပါမယ်။

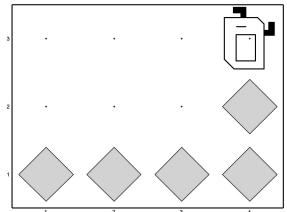
ပရီဂရမ် တစ်ခုလုံး အတွက်ဆုံးရင် အခုလိုပါ



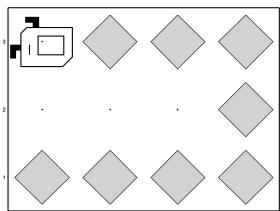
(က) မူလ အနေအထား



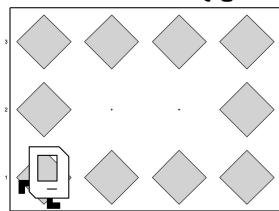
(ခ) ပထမ တစ်ကျွွဲပြီး



(ဂ) ဒုတိယ တစ်ကျွွဲပြီး



(ဃ) တတိယ တစ်ကျွွဲပြီး



(င) စုတွေဖြောက် ကျွွဲပြီး

ဗိ J·ဂ

```
# File: beeper_fence.py
from stanfordkarel import *
```

```
def main():
    for i in range(4):
        while front_is_clear():
            put_beeper()
            move()
        turn_left()

if __name__ == "__main__":
    run_karel_program("4x3")
```

Nested while and if

while နဲ့ if nest လုပ်လိုလည်း ရတာပေါ့။ လမ်းပေါ်မှာ ဘိပါတွေက ကြံရာကျပ်း ရှိနေမယ်။ ဘိပါတွေကို အမိုက်တွေလို ယူဆပြီး ရှင်းပေးပါမယ်။ လမ်းအလျားကို ကြံမသိဘူး၊ ကွန်နာတစ်ခုမှာ ဘိပါတစ်ခုထက် ပိုမိုရှိနိုင်ဘူးလို ယူဆပါ။ လမ်းအဆုံးထိ while loop နဲ့ ရွှေ့ခိုင်းလို ရမယ်။ ရောက်တဲ့ ကွန်နာတိုင်းမှာ ဘိပါရှိရင် ကောက်ခိုင်းပါမယ်။ if သုံးရမယ်။

```
from stanfordkarel import *
```

```
def main():
    while front_is_clear():
        if beepers_present():
            pick_beeper()
        move()
```

```

if beepers_present():
    pick_beeper()

if __name__ == "__main__":
    run_karel_program("clean_the_street")

```

if စတိတ်မန့် နဲ့ move ကို while loop ရဲ့ ဘလောက်ထဲမှာ ထည့်ပြီး ရွှေမျှရှင်းနေသ၍ ပြန်ကျော်ခိုင်း
ထားတာပါ။ if စတိတ်မန့်က ဘိပါရှိတော့မှာပဲ ကောက်ပေးဖို့အတွက်။

ပြီးခဲ့တဲ့ဟာနဲ့ အလားတူတဲ့ နောက်ထပ် ဥပမာ တစ်ခုကတော့ လမ်းတစ်လျှောက် ကွန်နာတွေမှာ ဘိ
ပါရှိရင် ကောက်ခိုင်းပြီး မရှိရင် တစ်ခုချေပေးရပါမယ်။ တန်ည်းအားဖြင့် ကွန်နာတစ်ခုချင်းရဲ့ ဘိပါရှိ/မရှိ
အခြေအနေကို ဆန့်ကျင်ဘက် ပြောင်းတာပေါ့။

```

# File: toggle beepers.py
from stanfordkarel import *

def main():
    while front_is_clear():
        if beepers_present():
            pick_beeper()
        else:
            put_beeper()
        move()

        if beepers_present():
            pick_beeper()

if __name__ == "__main__":
    run_karel_program("toggle beepers")

```

if...else သုံးထားတယ်။ ကျွန်ုတ်အားလုံး လမ်းရှင်းတဲ့ ပရိုဂရမ်နဲ့ တူတူပဲ။

Nested if

အခြေအနေ တစ်ရပ် မှန်တော့မှာပဲ လုပ်ချင်ရင် if ကို သုံးတယ်။ အခြေအနေ ‘နှစ်ရပ်’ လုံးနဲ့ ကိုက်ညီမှ
လုပ်ဆောင်စေချင်တဲ့အခါ nested if သုံးနိုင်ပါတယ်။ ညာဘက်လည်းရှင်း လက်ရှိကွန်နာမှာလည်း ဘိပါ
မရှိမှ ဘိပါတစ်ခု ချထားခိုင်းမယ်ဆိုပါစို့။ အခုလိုရေးနိုင်ပါတယ်

```

if right_is_clear():
    if no beepers_present():
        put_beeper()

```

ညာဘက်မှာ ရှင်းနေမှ အပြင် if က အတွင်း if ကို လုပ်ဆောင်စေမှာပါ။ အတွင်း if ကလည်း ဘိပါမရှိမှ
ပဲ put_beeper လုပ်မှာပါ။ ညာဘက်မှာ ပိတ်နေရင်သော်လည်းကောင်း ဘိပါရှိနေရင်သော်လည်းကောင်း
pick_beeper လုပ်မှာ မဟုတ်ပါဘူး။ right_is_clear နဲ့ no_beepers_present အခြေအနေ နှစ်
ခုလုံး မှန်မှာပဲ လုပ်မှာပါ။

ပုံ (၂.၈) (က) မှာ ဒုတိယ လမ်းတစ်လျှောက် လမ်းပြင်တဲ့အလုပ်ကို ကားရဲ့လှု တာဝန်ပေးတယ် လို့ ယူဆပါ။ ဘိပါရော ညာဘက်နံရံပါ မရှိတဲ့ ကုန်နာတွေက လမ်းအခြေအနေ တော်တော်ဆိုးနေတဲ့ နေရာ တွေ။ ဒီလို့နေရာတွေမှာ ဘိပါတစ်ခဲ ဖြည့်ပြီး လမ်းပြင်ပေးရမှာပါ။

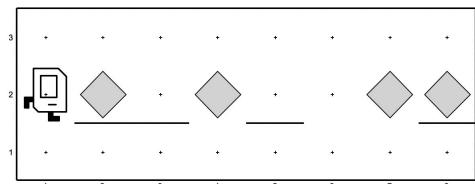
```
# File: repair_street.py
from stanfordkarel import *

def main():
    while front_is_clear():
        if right_is_clear():
            if no beepers_present():
                put_beeper()
                move()

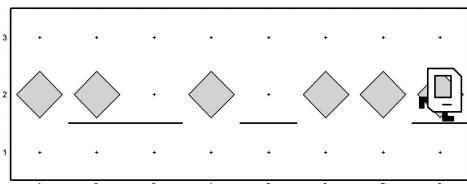
        if right_is_clear():
            if no beepers_present():
                put_beeper()

if __name__ == "__main__":
    run_karel_program("repair_street")
```

while ဘလောက်ထဲမှာ nested if နဲ့ move ပါဝင်တယ်။ nested if က ညာဘက်လည်းရင်း ဘိပါလည်းမရှိမှ လက်ရှိကွန်နာမှာ ဘိပါတစ်ခဲ ချခိုင်းထားတာပါ။ while ထဲမှာ if၊ အဲဒီ if ထဲမှာမှ နောက်ထပ် if တစ်ခဲ ဆင့်ထားတဲ့အတွက် သုံးဆင့် nest လုပ်ထားတာပါ။ အင်ဒန်ထဲ လုပ်ထားတာတွေ ဂရိုစိုက်ကြည့်ဖို့ လိုတယ်။ ဘယ်ဘလောက်ထဲမှာ ဘာရှိနေလဲဆိုတာ မြင်အောင် လုပ်ရမှာပါ။ ကိုယ်တိုင် ရေးရွင်လည်း အင်ဒန်ထဲ ဂရိုစိုက်ပြီး လုပ်ရပါမယ်။ လမ်းပြင်ပြီး အခြေအနေကို ညာဘက် ပုံ (၂.၈) (ခ) တွင် ကြည့်ပါ။



(က) မူလ အနေအထား



(ခ) လမ်းပြင်ပြီး အနေအထား

ပုံ ၂.၈

အခန်း ၃

ဖန်ရှင်များ (Functions)

ဖန်ရှင် (function) တွေဟာ ပရိုကရမ်းမင်းမှာ အရေးကြီးဆုံး အခြေခံသဘောတရားတစ်ခု ဖြစ်တယ်။ ဖန်ရှင်ဆိုတာ ဘာလဲ၊ ဘာကြောင့် အရေးပါရတာလဲ၊ ဖန်ရှင်တွေကို ပရိုကရမ် ဒီမိုင်းပြုလုပ် ရေးသားတဲ့အခါ ဘယ်လိုအသုံးချတာလဲ စတာတွေကို ဒီအခန်းမှာ လွှဲလာကြပါမယ်။

၃.၁ ဖန်ရှင် သတ်မှတ်ခြင်း

ညာဘက် လှည့်ခိုင်းချင်တိုင်း turn_left သုံးခါရေးနေရတာ ရေရှည်အဆင်မပြေပါဘူး။ turn_right လိုပဲ တိုက်ရိုက် ရေးလိုရှင် ပိုပြီးတော့ အဆင်ပြေမှာပါ။ ဒီလို လိုအပ်ချက်မျိုးကို ဖြည့်ဆည်း ပေးဖို့အတွက် ဟာ ဖန်ရှင်တွေရဲ့ အဓိက ရည်ရွယ်ချက်တွေထဲက တစ်ခုဖြစ်တယ်။ turn_right ဖန်ရှင်ကို အခုလို သတ်မှတ်နိုင်ပါတယ်။

```
def turn_right():
    turn_left()
    turn_left()
    turn_left()
```

ဒီလို သတ်မှတ်ထားပြီးရင် ညာဘက်လှည့်ချင်တဲ့အခါ

```
| turn_right()
```

လို တိုက်ရိုက်ပြေလို ရသွားမှာ ဖြစ်ပါတယ်။ turn_left သုံးခါ ရေးဖို့ မလိုတော့ပါဘူး။

ဖန်ရှင်တစ်ခု သတ်မှတ်တယ် (defining a function) ဆိုတာ စတိတ်မန်တွေကို ယူနစ်တစ်ခု အဖြစ် ဖွဲ့စည်းထားလိုက်တာပါပဲ။ ငြင်းယူနစ်အတွက် အမည်တစ်ခုကိုလည်း သတ်မှတ်ပေးတယ်။ ဖန်ရှင် သတ်မှတ်ချင်ရင် def စကားလုံးကို သုံးရပါတယ်။

အထက်ပါ turn_right ဖန်ရှင် သတ်မှတ်ချက် (function definition) မှာ

```
| def turn_right():
```

ကို ဖန်ရှင် ဟက်ဒါ (function header) လို ဆော်တယ် (မြန်မာလို ဆိုရင်တော့ ဖန်ရှင် ခေါင်းစည်းပေါ့)။ turn_right က ဖန်ရှင် အမည်။ ပါရာမီတာပါတဲ့ ဖန်ရှင်ဆိုရင် ပိုက်ကွင်းထဲမှာ ပါရာမီတာတွေ သတ်မှတ်ရတယ်။ ဥပမာ (x, y)။ ပါရာမီတာ မပါရင်တော့ () ပဲဖြစ်မယ်။ ကားရဲလှုံးမှ ဖန်ရှင်အားလုံး

ဟာ ပါရာမီတာ မပါတဲ့အတွက် () ပဲ ဖြစ်မှာပါ။ ဖန်ရှင်ဟက်ဒီ လိုင်းအဆုံးမှာ ကော်လံ ‘:’ ထည့်ပေး ဖို့ လိုပါတယ်။

ဖန်ရှင် ဟက်ဒီအောက် အင်ဒန်ထုပ်ထားတဲ့ လိုင်းအားလုံးဟာ ငြင်းဖန်ရှင်နဲ့ သက်ဆိုင်တဲ့ ကုဒ် ဘလောက် ဖြစ်တယ်။ အခါ turn_right ဖန်ရှင် ဘလောက်မှာ turn_left သုံးကြိမ်ပါတယ်။

| turn_right()

လုပ်ခိုင်းတာက (သတ်မှတ်ထားတဲ့) ဖန်ရှင်ကို အသုံးပြုတာ ဖြစ်တယ်။ ဒီအခါမှာ ငြင်းဖန်ရှင်နဲ့ သက်ဆိုင်တဲ့ ဘလောက်ကို လုပ်ဆောင်ပေးမှာပါ။ ဖန်ရှင်ကို အသုံးပြုတာကို ဖန်ရှင်ကောင် (function call) လုပ်တယ်လို့ ပြောပါတယ်။

ဖန်ရှင်သတ်မှတ်တာနဲ့ ဖန်ရှင်ကောင် လုပ်တဲ့ပုံစံကို အောက်ပါအတိုင်း ယေဘုယျအားဖြင့် တွေ့ရပါမယ်။ Python ထုံးစာရေ ဖန်ရှင်နဲ့မည်မှာ စာလုံးအသေးကိုပဲ ထုံးလေ့ရှိတယ်။ စကားလုံး နှစ်ခုနဲ့ အထက်ဆိုရင် ကြားမှာ underscore (_) ဥားပေးလေ့ ရှိတယ်။

| def name_of_function():
 statement1
 statement2
 statement3 etc.

| name_of_function()

ဖန်ရှင်နဲ့မည် အဓိပ္ပာယ် အရေးကြီးပါတယ်

စာရေးတာပဲဖြစ်ဖြစ်၊ ပရိုဂုရမ်ကုဒ် ရေးတာပဲဖြစ်ဖြစ် စိတ်ထဲ တေးတဲ့အတိုင်း၊ စဉ်းစားတဲ့အတိုင်း ပေါ်လွှာအောင် ဖော်ပြနိုင်တာဟာ အားသာချက်တစ်ခုပါပဲ။ ဖန်ရှင် သတ်မှတ်ထားခြင်း အားဖြင့် ညာဘက်လှည့်ခိုင်းရင် turn_right ဘိပါ နှစ်ဆယ့်ငါးခု ချုပ် put_25_beeper တိုက်ရှိက် ဖော်ပြလို့ ရတာဟာ အရေးပါတဲ့ ကိစ္စဖြစ်ပါတယ်။ ဘာသာစကားတစ်ခုရဲ့ ဖော်ပြနိုင်စွမ်း ‘အား’ (expressive power) ကို ထပ်လောင်းအားဖြည့်ပေးတာလို့ ဆိုရမှာပါ။

ဖန်ရှင်လုပ်ဆောင်ပေးတဲ့ ကိစ္စကို သိသာမေ့မဲ့ နားလည်ရလွှယ်မဲ့ နံမည်မျိုး ကရှိက်ရွေးချယ်တာက လည်း အရေးပါပါတယ်။ ကားခဲ့လုပ်ပရိုဂုရမ်တွေမှာ အမိန့်ပေးခိုင်းစေတဲ့ ပုံစံနဲ့ ဖန်ရှင်နဲ့မည်ပေးလေ့ရှိတယ်။ ဥပမာ turn_north, pick_all_beeper ။

ဖန်ရှင်သတ်မှတ်ချက်နဲ့ ဖန်ရှင်ကောင် ဥပမာ တချို့ကို လေ့လာကြည့်ပါ။ ဘိပါ နှစ်ဆယ့်ငါးခု ချေပေးတဲ့ put_25_beeper ဖန်ရှင်ပါ

| def put_25_beeper():
 for i in range(25):
 put_beep()

| put_25_beeper()

ဒါကတော့ ကွန်နာတစ်ခုမှာ ရှိတဲ့ ဘိပါအားလုံးကောက်ပေးတဲ့ ဖန်ရှင်ဖြစ်ပါတယ်

| def pick_all_beeper():
 while beepers_present():
 pick_beep()

```
|pick_all_beeper()
```

၃.၂ ဖန်ရှင် Composing ဖန်ရှင်

`pickAllBeepers` မက်သွားလာ ကွန်နာတစ်ခုမှာရှိတဲ့ ဘိပါအားလုံးကိုကောက်ပေးပါတယ်။ ဘိပါမရှိတဲ့ ကွန်နာမှာလည်း သုံးလို့ရတယ်။ တစ်ခုနဲ့အထက်ရှိရှင်တော့ အားလုံးကွန်တဲ့ထိ ကောက်ပေးမှာပါ။

```
void pickAllBeepers() {
    while (beepersPresent()) {
        pickBeeper();
    }
}
```

`pickAllBeepers` ကို အခြေခံအစိတ်အပိုင်းတစ်ခုအနေနဲ့ အသုံးပြုပြီး လမ်းတစ်လျှောက် ကွန်နာ တွေအားလုံးက ဘိပါတွေကို ရှင်းပေးမဲ့ `cleanTheStreet` မက်သွားကို သတ်မှတ်နိုင်ပါတယ်။

```
void cleanTheStreet() {
    while (frontIsClear()) {
        pickAllBeepers();
    }
    pickAllBeepers();
}
```

အခြေခံကျြီး ရှိုးရှင်းတဲ့ အစိတ်အပိုင်းလေးတွေကနေ ပို့ပြီးရှုပ်ထွေးခက်ခဲတဲ့ ကိစ္စတွေကို ဖြေရှင်းဆောင်ရွက်ပေးနိုင်တဲ့ မက်သွားတွေဖြစ်လာအောင် ဖွဲ့စည်းတည်ဆောက်လို့ရတာကို တွေ့ရပါတယ်။ ဒီလိုနည်းနဲ့ မက်သွားတွေ ဖွဲ့စည်းတည်ဆောက်တာကို Method Composition လိုခေါ်ပါတယ်။ Composition ကို တစ်ဆင့်ပြီးတစ်ဆင့် လုပ်လို့ရတယ်။ ကားရဲကမ္ဘာထဲရှိ ကွန်နာအားလုံးက ဘိပါတွေကို ရှင်းပေးမဲ့ `cleanTheWorld` မက်သွားလာတဲ့အခါ `cleanTheStreet` ကို အခြေခံအစိတ်အပိုင်းအဖြစ် အသုံးပြုနိုင်မှာ ဖြစ်တယ်။ (လမ်းတွေအားလုံး ရှင်းတာဟာ ကားရဲကမ္ဘာထဲရှိ ကွန်နာအားလုံးကို ရှင်းတာပါပဲ)

`pickAllBeepers` က အရှိုးရှင်းဆုံး အခြေခံအကျဆုံး ဖြစ်တယ်။ ဒါကိုအသုံးပြုပြီး အတန်အသင့် ပို့ရှုပ်ထွေးတဲ့ `cleanTheStreet` ကို တည်ဆောက်တယ်။ တစ်ခါ `cleanTheStreet` ကို အခြေခံပြီး `cleanTheWorld` ကို ဆက်လက်တည်ဆောက်တယ်။ ဒီလိုနည်းလမ်းနဲ့ ရှိုးရှင်းတဲ့ အစိတ်အပိုင်းလေးတွေကနေ ပို၍ပို၍ ကြိုးမားရှုပ်ထွေးတဲ့ အစိတ်အပိုင်းတွေကို တစ်ဆင့်ပြီးတစ်ဆင့် တည်ဆောက်ယူလို့ရနိုင်တာ `pick` `pick` ကို တွေ့ရပါတယ်။

၃.၃ ဖန်ရှင်တွေ ဘာကြောင့် အရေးပါရတာလ

```
def pick_all_beeper():
    while beepers_present():
        pick_beeper()
    pick_all_beeper()
```


နောက်ဆက်တဲ့ က

လိုအပ်သည့် ဆောဖံ့ဌများ ထည့်သွင်းခြင်း

အခြား အင်ဂျင်နီယာ/သိပ္ပံဌ ပညာရပ်တွေလိုပဲ ပရိုကရမ်မင်းလေ့လာတဲ့အခါ လက်တွေ့လုပ်ကြည့်ဖို့ အင်မတန်အရေးကြိုးပါတယ်။ လက်တွေ့ရေးမကြည့်ဘဲ စမ်းသပ်မကြည့်ဘဲ သဘောတရားပိုင်းဆိုင်ရာတွေကို အမှန်တကယ်နားလည်ဗူး မဟုတ်ပါဘူး။ အခြားပညာရပ်တွေထက် ကွန်ပျိုးတာ ပရိုကရမ်မင်းရဲ့ အားသာချက်တစ်ခုကတော့ လက်တွေ့စမ်းသပ်ခန်းကြိုးတွေ ရှိစရာမလိုတာပါပဲ။ စမ်းသပ်ပစ္စည်းတွေ လည်း များများစားစား မလိုအပ်ဘူး။ ကွန်ပျိုးတာ တစ်လုံးနဲ့ လိုအပ်တဲ့ ဆောဖံ့ဌပဲတဲ့ ရှိရင်ရပြီ။ ဆောဖံ့ဌတွေကလည်း ပိုက်ဆံကုန်စရာမလိုဘူး။ မပေးဘဲ သုံးလို့ရတာ။

ပရိုကရမ်လက်တွေ့ရေး လေ့ကျင့်ဖို့အတွက် လိုအပ်တဲ့ ဆောဖံ့ဌတွေ ထည့်ထားရပါမယ်။ Python programming language အတွက် Python ဆောဖံ့ဌ ထည့်ရပါမယ်။ Python ဆောဖံ့ဌမရှိဘဲ Python ကုဒ်တွေ၊ Python ပရိုကရမ်တွေ run လို မရပါဘူး။ Python အပြင် ပရိုကရမ်ကုဒ် ရေးဖို့ အတွက် အထောက်အကူဗြာ ဆောဖံ့ဌပဲတစ်ခုလည်း လိုအပ်တယ်။

ပရိုကရမ် ကုဒ်ရေးဖို့အတွက် PyCharm သိမဟုတ် Visual Studio Code (VS Code) ကို အသုံးပြုနိုင်ပါတယ်။ အက်ဆေးတစ်ပုဒ်ရေးတဲ့အခါ မိုက်ခရိုဆောဖံ့ဌ Word နဲ့ရေးလိုရသလို ရိုးရိုးရှင်းရှင်း Notepad လောက်နဲ့ ရေးလိုလည်း ရတာပါပဲ။ အက်ဆေးရဲ့ အဓိပ္ပာယ်ကသာ အစိုကပါ။ ပရိုကရမ်ကုဒ် ရေးတဲ့အခါမှာလည်း ခီးသောပါပဲ။ ဆောဖံ့ဌပဲရေးကြိုးတွေ တည်ဆောက်တဲ့အခါ လိုအပ်မဲ့ ဖီချာတွေ အားလုံး စုံစုံလင်လင်ပါပြီးသား PyCharm လို Integrated Development Environment(IDE) ဆောဖံ့ဌပဲများနဲ့ ကုဒ်တွေရေးလိုရသလို ပါ့ပေါ့ပါ့ပါ့နဲ့ လိုအပ်မှုပဲ လိုတဲ့ဖီချာအတွက် extensions (plugin လိုလည်းခေါ်ယ်) ထည့်သွင်းရတဲ့ Visual Studio Code (VS Code) လို ကုဒ်အသိဒ်ဘာ (Code Editor) ဆောဖံ့ဌပဲ့ သုံးပြီး ရေးရင်လည်း ရတာပါပဲ။ နောက်ဆုံး ကုဒ်နည်နည်း၊ ဖိုင်နည်းနည်း ရိုးရှင်းတဲ့ ပရိုကရမ်လေးတွေဆိုရင် Notepad နဲ့ ရေးလိုတောင် ရပါတယ်။ ကုဒ်တွေများမယ် ဖိုင်တွေ များမယ်၊ အသင်းအစွဲလိုက် ပူးပေါင်းရေးရတဲ့ ပရိုကရမ်မျိုးတွေ ဆိုရင်တော့လည်း Notepad လောက်နဲ့ အဆင်မပြနိုင်တော့ဘူးပေါ့။

PyCharm ရော VS Code ထည့်သွင်းနည်းပါ ဖော်ပြပေးပါမယ်။ မိမိ နှစ်သက်ရာ အဆင်ပြောရာ သို့မဟုတ် နီးစပ်ရာ ပရိုကရမ်ဘာ အသိမိတ်ဆွေ အကြုပြုတဲ့ တစ်ခုကို ရွေးချယ်သုံးပါ။ စလေ့လာသူအနေနဲ့ PyCharm ကို အသုံးပြုတာ ပိုအဆင်ပြောမယ်လို ထင်တယ်။ PyCharm သုံးကြည့်လို မိမိကွန်ပျိုးတာ မှာ နေးလွန်းတယ်ဆိုရင် VS Code ကိုစမ်းကြည့်ပါ။ နှစ်ခုလုံး စက်အရမ်းကောင်း/မြင့် ဖို့ မလိုပါဘူး။ တော်ရုံး အတန်အသင့်ကောင်းတဲ့ စက်လောက်နဲ့ အဆင်ပြောပါတယ်။

မိမိကိုယ်တိုင်က ကွန်ပျိုးတာ အသုံးပြုပဲ အခြေခံ အားနည်းပြီး ဖော်ပြပေးထားတဲ့ အတိုင်း တစ်ဆင့် ချင်း အင်စတောလ် လုပ်ဘာလည်း အဆင်မပြောစွာရင် ဒီဘာအုပ်ရဲ့ အောက်ပါ ဖွံ့ဖြိုးတွေနဲ့ ယူကျ။

ချုပ်နယ်တွေမှာ ကြည့်ရှုမေးမြန်း အကူအညီ တောင်းဆိုင်ပါတယ်။ ဒါမှမဟုတ် အတွေးအကြိုးစိတ် နဲ့ စီးပွားရေးမှုပြီး အင်စတောလ်လုပ်ပါ။

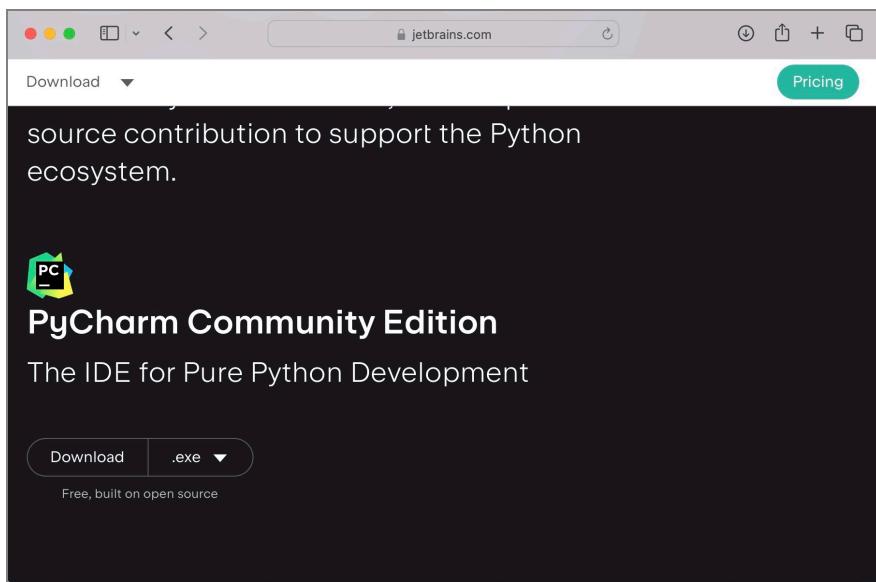
<https://www.facebook.com/bpwp>

<https://www.youtube.com/bpwp>

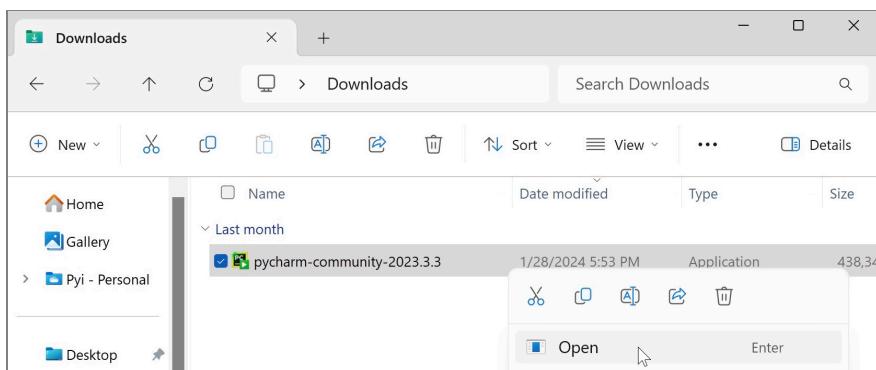
PyCharm အတွက် အရင်ဖော်ပြပေးပါမယ်။ VS Code အတွက် စာမျက်နှာ ငါးမွှာ ကြည့်ပါ။

Python နှင့် PyCharm IDE ထည့်သွင်းခြင်း

<https://www.jetbrains.com/pycharm/download/> လင့်ကိုဖွင့်ပါ။ ဝဘ်စာမျက်နှာ အောက်ဘက် နည်းနည်း ဆွဲချက်ရင် PyCharm Community Edition ခေါင်းလုပ်ခလုပ်ကို တွေ့ရပါမယ်။ ပုံ (က/ခ) ကိုကြည့်ပါ။



ပုံ ၂/၁၀



ပုံ ၃/၂၂

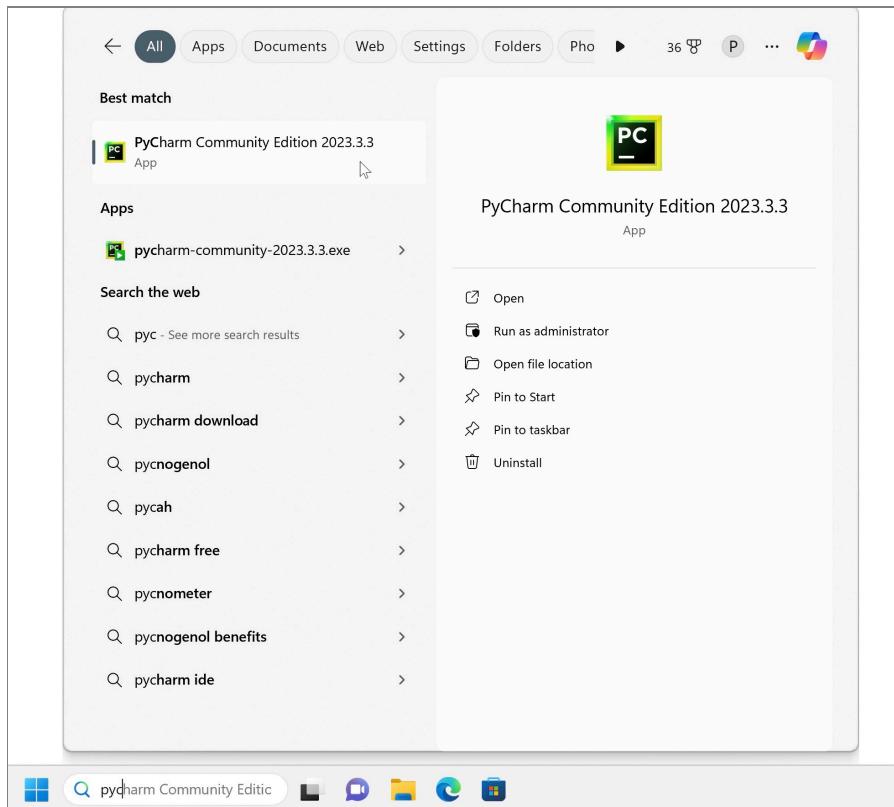
PyCharm Community Edition ကို အောင်လုပ်လုပ်ပါ။ (ဝဘ်စာမျက်နှာ အပေါ်ပိုင်းက ၀၂၍ သုံးရတဲ့ PyCharm Professional ကို အောင်လုပ် မှားမလုပ်မိန့် သတိပြုပါ)။ အင်စတော်လာဖိုင်ကို ညာကလစ်နှုပ်ပြီး Open လုပ်ပါ (ပုံ ၂/၂ ကိုကြည့်ပါ)။ Yes/No မေးတဲ့အခါ Yes နှုပ်ပါ။ Next > ကို နှုပ်၍ ရွှေ့ကိုဆက်သွားပြီး နောက်ဆုံးမှာ Install နှုပ်ပြီး ကွန်ဖန်လုပ်ပါ။ အင်စတော်လာပြီးသွားရင် Finish နှုပ်ပါ။

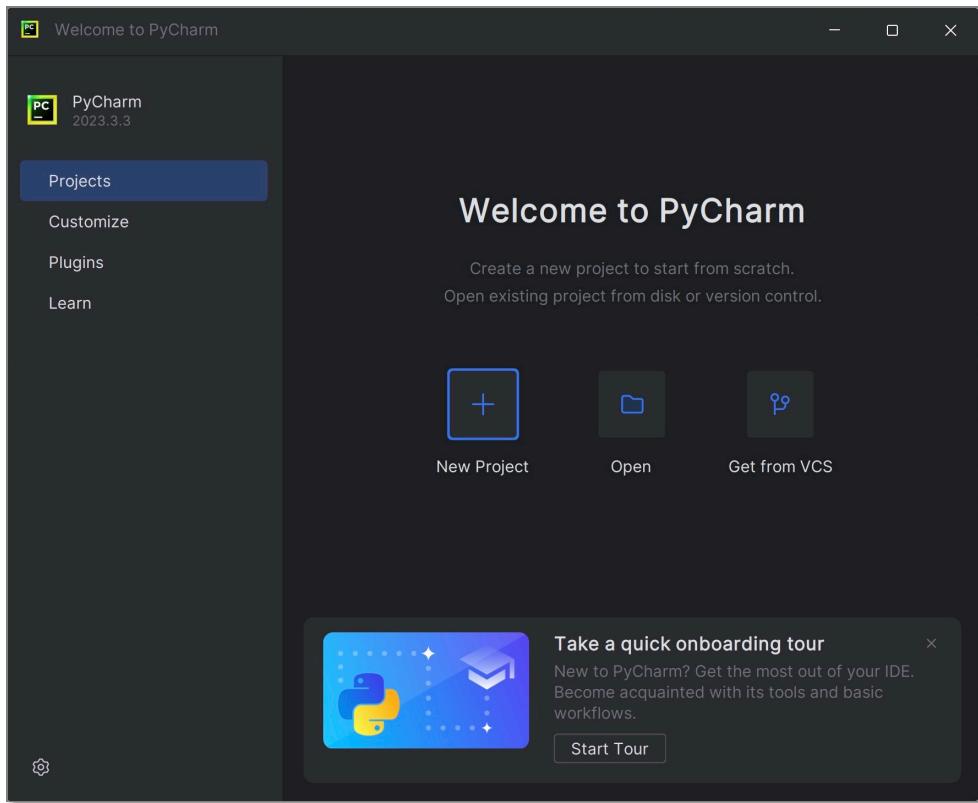
၀၂:၃၅ Taskbar Search ကနေ PyCharm ကိုရှုပြီးဖွံ့ဖြိုးပါ (ပုံ ၂/၃ ကို ကြည့်ပါ)။ သဘောတူ ကြောင်း ကွန်ဖန်လုပ်ခိုင်းရင် ချက်ချက်ဘောက်စ် ချက်ချက်လုပ်ပြီး Continue နှုပ်ပါ။ ဒေတာပို့ချင်လား ထပ် မေးပါလိမ့်မယ်။ Don't Send နှုပ်ပါ။ Welcome စာရင်ကိုပေါ်လာမယ်။ ပုံ (၂/၄) မှာ ကြည့်ပါ။

ဒီစာရေးနေချိန် လက်ရှိ PyCharm ဗားရှင်းက ၂၀၂၃ ပါ။ သိပ်မကြာခင် ၂၀၂၄ ထွက်ပါတော့မယ်။ အကယ်၍ လက်ရှိဟားရှင်းထက် နိမ့်တဲ့ဟားရှင်းတွေကို အောင်လုပ် လုပ်ချင်ရင် အောက်ပါ လင့်ကို သွားပါ။

<https://www.jetbrains.com/pycharm/download/other.html>

ဗားရှင်း ၂၀၂၄/၂၅ ထွက်ပြီးတဲ့ အချိန်မှ ၂၀၂၃ ဗားရှင်းကို လိုချင်ရင် ဝဘ်စာမျက်နှာမှ ရှာပြီး အောင်လုပ်လုပ်ပါ။ ၂၀၂၃ မှာလည်း ဗားရှင်းအခွဲတွေ ရှိပါသေးတယ်။ လက်ရှိအမြင့်ဆုံး ဗားရှင်းအခွဲ (ဥပမာ ၂၀၂၃.၃.၃) ကို သုံးလိုပါတယ်။ ၂၀၂၄/၂၅ သုံးမယ်ဆုံးရင်လည်း ပြဿနာတွေ မရှိပါဘူး။ Update ဗားရှင်းဖြစ်တဲ့အတွက် ပုံတွေမှာ ပြထားတာနဲ့တော့ ကွားချက်တရာ့၏ ရှိကောင်းရှိနှင့်ပါတယ်။

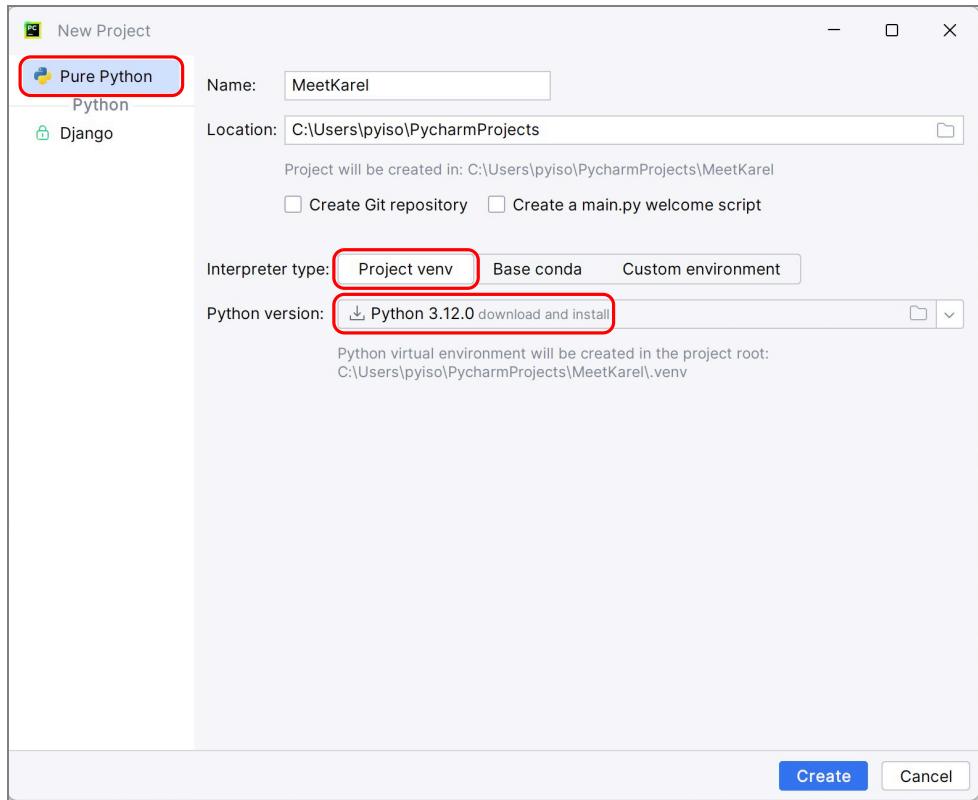




ပုံ ၂/၄

PyCharm IDE ဆိတာဘာလ

PyCharm ဟဲ Python နဲ့ ဆောင်ဝဲ ရေးဖိုအတွက် အထောက်အကူဗြှု Integrated Development Environment(IDE) ဆောင်ဝဲဖြစ်ပါတယ်။ စာစီတရုက်လုပ်တဲ့အခါ Microsoft Word ကို အသုံးပြု ကြသလိုပဲ Python ကုဒ်ရေးနဲ့ PyCharm ကိုထံးတဲ့ သဘောပေါ့။ PyCharm IDE ၏ Python ပရောဂျက်တွဲ အတွက် အခိုက်ရည်ရွယ်တယ်။ အထောက်အဦးတစ်ခု၊ တံတားတစ်ခု ဆောက်လုပ်တာ ကို ပရောဂျက်လို့ ပြောလေ့ရှိသလို ပရိုကရမ်/ဆောင်ဝဲတစ်ခု တည်ဆောက်တာကိုလည်း ပရောဂျက်လိုပဲ သုံးနှုန်းပါတယ်။ တစ်ဦးတစ်ယောက်တည်း ရေးတဲ့ ပရိုကရမ်အသေးလေးတွဲ အတွက် PyCharm ကို အသုံးပြုနိုင်သလို ပရိုကရမ်မာတွေ အဖွဲ့လိုက်နဲ့ တည်ဆောက်ရတဲ့ ပရောဂျက်ကြီးတွဲ အတွက်လည်း သုံးပါတယ်။ ဒီစာအုပ်မှာတော့ PyCharm ရဲ့ အဆင့်မြင့်ဖိုချာတွေကို အသုံးပြုမှာ မဟုတ်ပါဘူး။ ပရိုကရမ်းမင်း စလေ့လာသူတွေကို လွှာယ်ကူးအဆင်ပြေစေတဲ့ အခြေခံ ဖိုချာတွေလောက်ပဲ အသုံးပြုမှုပါ။



ပုံ ၂/၅ ပရောဂျက် အသစ်ယူခြင်း

PyCharm ပရောဂျက်ဆောက်ခြင်း

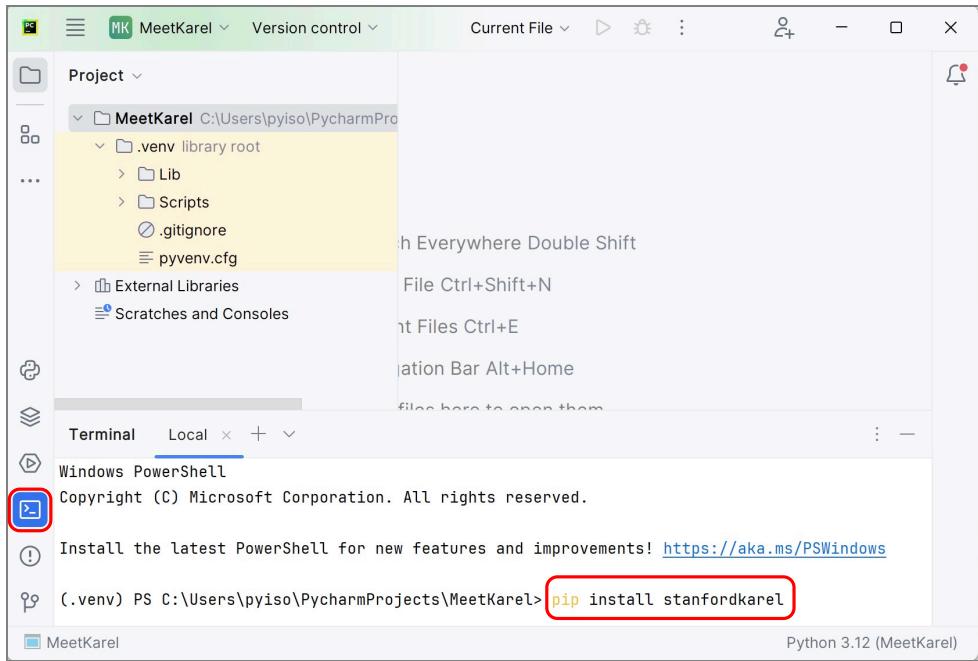
အင်စတောလုပ်ပြီးရင် PyCharm IDE ကိုဖွံ့ဖြိုး Welcome စခရင်မှ သို့မဟုတ် File မိန္ဒီးမှ New Project နှင့်ပြီး ပရောဂျက် အသစ်ယူပါ (ပြီးခဲတဲ့ ပုံ ၂/၅ ကို ပြန်ကြည့်ပါ)။ နံမည်ကို MeetKarel သေးပါ။ ပုံ (၂/၅) မှာ တွေ့ရတဲ့အတိုင်း Pure Python, Proj venv နဲ့ Python ဘားရှင်း 3.12.xx ကို ရွေးပါ။ xx က အမြားကဏ္ဍား ဖြစ်နေနိုင်တယ်။ အဓိက ဘားရှင်း 3.12 သာ ဖြစ်ပါပေါ်။ Create ခလုတ်နှင့်ပါ။ ပရောဂျက်အတွက် Python ကို အင်စတောလ လုပ်ပါလိမ့်မယ်။

မိမိလေ့လေနေတဲ့ အခန်းတစ်ခန်းချင်းစီအတွက် ပရောဂျက်တစ်ခဲ ဆောက်နိုင်ပါတယ်။ အခန်း (၂) အတွက် Chapter02၊ အခန်း (၃) အတွက် Chapter03 စသည်ဖြင့်။ သက်ဆိုင်ရာအခန်းအလိုက် ကုဒ်ဖိုင်တွေကို ပရောဂျက်တစ်ခဲစီမှာ ထားတဲ့အတွက် ဖိုင်တွေများပြီး ရှုပ်ထောင်းပွဲနေတဲ့ ပြဿနာ မရှိတော့ဘူး။ ကုဒ်ဖိုင်တွေကို ပရောဂျက်တစ်ခဲတဲ့မှာပဲ ဖိုဒါအလိုက်ခဲ့ထားလို့ရပေမဲ့ စလေ့လာသူအနေနဲ့ ပရောဂျက်တစ်ခဲချင်း ခဲ့ထားတာလောက် မလွယ်ကူဘူး။

ပရောဂျက် အသစ်ယူတဲ့အခါ တည်နေရာ Location: ကို သူ့ရှိခိုးအတိုင်း ထားနိုင်သလို မိမိထားချင်တဲ့နေရာကို ညာဘာက်စွန်း ဖိုဒါအိုင်ကွန်လေးနှင့်ပြီး ပြောင်းလို့ရတယ်။

stanfordkarel လိုက်ဘရီ အင်စတောလုပ်ခြင်း

ပုံ (၂/၆) မှာ အနီရောင် ဝိုင်းပြထားတဲ့ အိုင်ကွန်ကို နှင့်ပြီး Terminal ကိုဖွံ့ဖြိုး။ Terminal မှာ အောက်ပါ ကွန်မန်းဖြင့်



ပုံ ၂/၆ Karel လိုက်ဘရီ အင်စတောလ်လုပ်ခြင်း

```
pip install stanfordkarel
```

ကားရဲလ်လိုက်ဘရီကို အင်စတောလ်လုပ်ပါ။ ပုံ (၂/၆) မှာ အနီရောင် စိုင်းပြထားပါတယ်။ ခက္ကာတဲ့ အခါ အောင့် မက်ဆွဲချုပ်တွေ ကျလာပါလိမ့်မယ်။

```

Collecting stanfordkarel
  Downloading stanfordkarel-0.2.7-py3-none-any.whl (51 kB)
    51.9/51.9 kB 443.1 kB/s
      eta 0:00:00
Installing collected packages: stanfordkarel
Successfully installed stanfordkarel-0.2.7

```

```

[notice] A new release of pip is available: 23.2.1 -> 24.0
[notice] To update, run: python.exe -m pip install --upgrade pip
(.venv) PS C:\Users\pyiso\PycharmProjects\MeetKarel>

```

ဟိုက်လိုက်ပြထားတဲ့ မက်ဆွဲချုပ် တွေ့ရရင် အင်စတောလ်လုပ်တာ အောင်မြင်လိုပါ။

မှတ်ချက်။ ပရောဂျက် အသစ်ဆောက်တိုင်း လိုအပ်တဲ့ လိုက်ဘရီကို တစ်ခါထပ်ပြီး အင်စတောလ်လုပ်ရပါမယ်။ ကားရဲလ်အခန်း ပရောဂျက်တစ်ခုစီအတွက် stanfordkarel လိုက်ဘရီကို အထက်ဖော်ပြပါအတိုင်း အင်စတောလ်လုပ်ဖို့လိုတယ်။

နမူနာ ကားရဲလ် ကဗ္ဗာနှင့် ပရိဂရမ်ကုဒ် ဖိုင်များထည့်ခြင်း

meet_karel.zip ဖိုင်ကို ဒီလင့် <http://tinyurl.com/3mmmm9c7j> ကနေ ဒေါင်းလုဒ်လုပ်ပါ။ ငါး zip ဖိုင်ကို extract လုပ်ပါ။ meet_karel နံမည်နဲ့ ဖိုဒါတစ်ခု ရလာပါမယ်။ ငါးဖိုဒါထဲမှ အောက်ပါ worlds ဖိုဒါနှင့် .py ဖိုင်အားလုံးကို ကော်ပီလုပ်ပါ။

- worlds
 - ▶ meet_karel.w
 - ▶ move_beeper_to_other_side.w
- meet_karel.py
- move_beeper_to_other_side.py
- world_editor.py

MeetKarel ပရောဂျက်ထဲတွင် ကူးထည့်ပါ။ ပင်မ ပရောဂျက် MeetKarel (ပုံ က/၇ မှာ မြှားပြထား) ပေါ်မှာ ညာကလစ်နှင့်ပြီး Paste လုပ်ရမှာပါ။ ကော်ပီကူးထည့်လိုက်တဲ့ ဖိုင်တွေက ပုံမှာ တွေ့ရတဲ့ အတိုင်း MeetKarel ဖိုဒါအောက်မှာ ရှိသင့်ပါတယ်။

အခန်းအလိုက် နမူနာ ကုဒ်ဖိုင်တွေ ထည့်ပေးထားတဲ့ .zip ဖိုင်တွေကိုလည်း အထက်ပါအတိုင်း အလားတူ လုပ်ရပါမယ်။ ပရောဂျက်အသစ်ဆောက်၊ .zip ဖိုင်ကို ဖြည့်၊ ရလာတဲ့ ဖိုဒါထဲက ဖိုင်တွေကို ပင်မ ပရောဂျက် ဖိုဒါထဲ ကော်ပီကူးထည့် ရုံပါပဲ။

meet_karel.py ဖိုင်ကို ကလစ်နှစ်ချက်နှင့် ဖွင့်ပါ။ ပုံ (က/၇) မှာ အနီးပိုင်းထားတဲ့ ကုဒ်အယ်ဒီတာ (code editor) ပွင့်လာပါမယ်။ အဲဒီ မူရင်းဖိုင်မှာ အောက်ပါအတိုင်း ဆက်လက် ပြင်ဆင်ဖြည့်စွာက်ပါ။

```
from stanfordkarel import *

def main():
    """Karel code goes here!"""
    move()
    move()
    move()
    pick_beeper()
    turn_left()
    move()
    move()

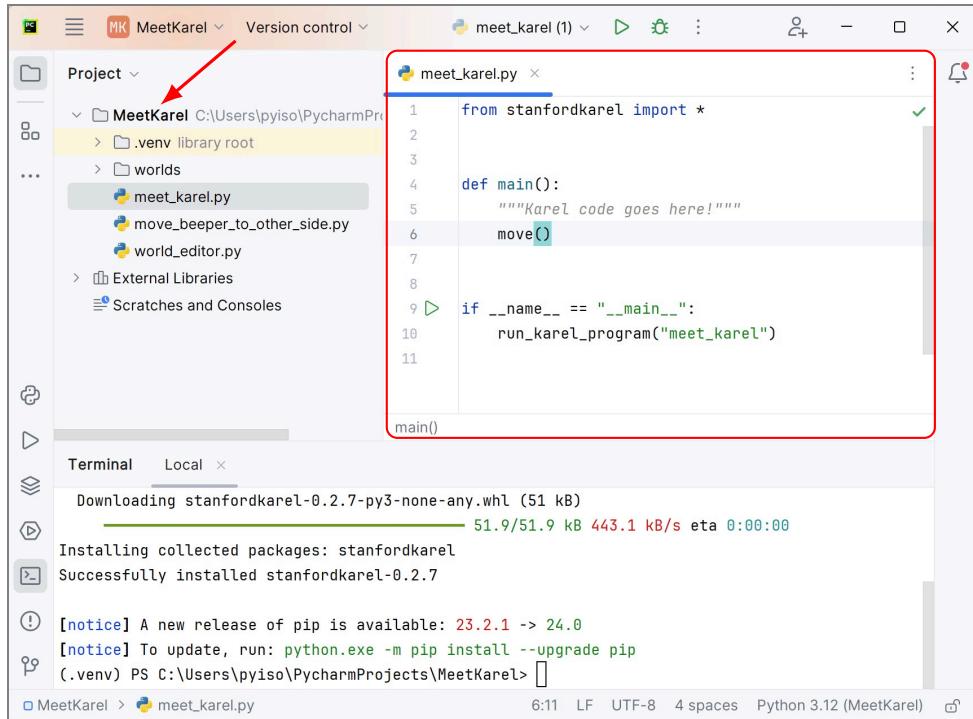
    turn_left()
    turn_left()
    turn_left()

    move()
    put_beeper()

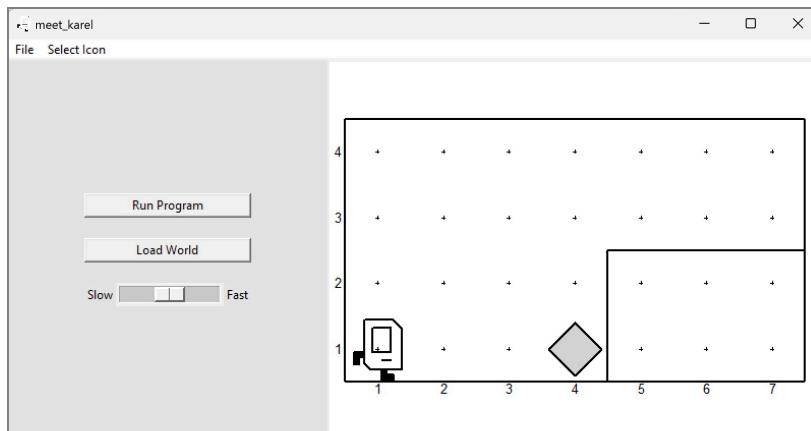
if __name__ == "__main__":
    run_karel_program("meet_karel")
```

run ထားတဲ့ ပရိဂရမ်တစ်ခုကို တစ်ခါထပ် run ရင် Cancel (သို့) Stop and Rerun လုပ်မှာလား မေးတယ်။ Stop and Rerun လုပ်ရပါတယ်။

meet_karel.py ဖိုင်ကို ညာကလစ်နှင့်ပြီး Run 'meet_karel' လုပ်ပါ (အယ်ဒီတာ ဧရိယာမှာ ညာကလစ်နှင့်ပြီး Run 'meet_karel' လုပ်လိုလည်း ရပါတယ်) ။ ပုံ (က/၈) က ကားရဲ့ပုံ ပရိဂရမ် တက်လာရင် Run Program ခလုတ်ကိုနိုပ်ပါ။ ဘိပါကို ကားရဲ့လ်က ရွှေပေးပါလိမ့်မယ်။



ပုံ ၂/၇



ပုံ ၃/၇

ဆင်းတက်စံအမှားများ

အကယ်၍ ပရိုက်မဲ့ run ပေါ်လို့မရရင် ကုဒ်ရေးတာမှားနေလို့ ဖြစ်နိုင်တယ်။ မမိရေးထားတာကို စာမျက်နှာ (၃၅) က ပရိုက်မဲ့ကုဒ်နဲ့ နှင့်ယူဉ် စစ်ဆေးကြည့်ပါ။ PyCharm အယ်ဒီတာမှာ အနိုင်တွန်းလေးတွေ (ပုံ က/၉) ပြတဲ့နေရင် အဲဒီနေရာတွေမှာ ဆင်းတက်စံမှားနေလို့ (သို့) လိုက်ဘရှိမထည့်ရသေးလို့ပဲ။

လိုက်ကွင်းကျို့နေတာက အဖြစ်များတဲ့ အမှားပါ။ ကျို့ခဲ့လို့ မရပါဘူး။ အင်ဒန်တေးရှင်း (indentation) လုပ်ရမဲ့နေရာမှာ မလုပ်ထားရင်လည်း ပြဿနာဖြစ်တယ်။ move, turn_left တွေကို ဘေးမျဉ်းလာဘက်ဆွဲပြီး အင်ဒန်လုပ်ပေးရမယ်။ အဲဒါတွေ ဂရာမစိုက်မိရင် ဆင်းတက်စံအမှားဖြစ်ပြီး ပရိုက်မဲ့ run လို့ မရနိုင်ဘူး။

Terminal မှာ ထုတ်ပေးတဲ့ မက်ဆွဲချုတွေကို ကြည့်ပြီးတော့လည်း ဘာပြဿနာဖြစ်နေလဲ မှန်းဆ လို့ရနိုင်တယ်။ ဘာကြောင့်ဖြစ်နိုင်လဲ ဆက်စပ်စဉ်းတားလို့ ရတယ်။ ဥပမာ ဖြစ်တဲ့ပြဿနာအလိုက် အခုလို တွေ့ရပါမယ်။

```
File "c:\Users\pyiso\VS Code\meet_karel\meet_karel.py", line 6
    move(
        ^
SyntaxError: '(' was never closed

File "c:\Users\pyiso\VS Code\meet_karel\meet_karel.py", line 7
    move()
        ^
IndentationError: unexpected indent

Traceback (most recent call last):
  File "c:\Users\pyiso\VS Code\meet_karel\meet_karel.py", line 1,
    in <module>
      from stanfordkarel import *
ModuleNotFoundError: No module named 'stanfordkarel'
```

The screenshot shows the PyCharm interface with the following details:

- Project:** MeetKarel
- File:** meet_karel.py
- Code Content:**

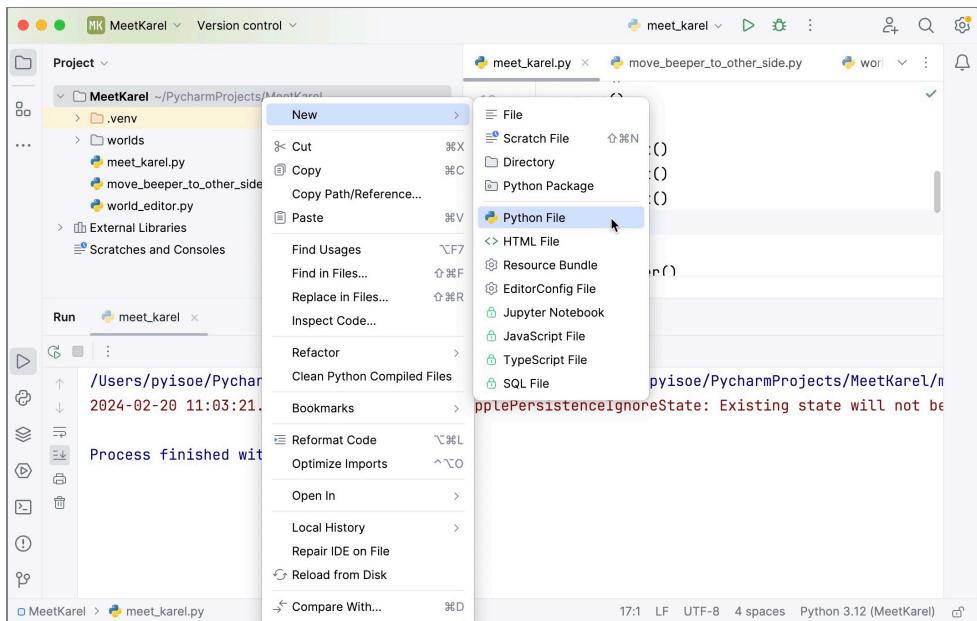
```
def main():
    """Karel code goes here!"""
    move()
    move()
    turn_left()

if __name__ == "__main__":
    pass
```
- Run Tab:** Run configuration for 'meet_karel'.
- Bottom Status Bar:** Shows the file path 'C:\Users\pyiso\PycharmProjects\MeetKarel\meet_karel.py', line numbers 9:1, LF, UTF-8, 4 spaces, Python 3.12 (MeetKarel), and a small icon.

Python ဖိုင် အသစ်ယူခြင်း

MeetKarel ပင်မ ပရောဂျက်ဖိုဒ်အပေါ်မှာ ညာကလစ်နှုပ်ပြီး Python ဖိုင် အသစ်ယူနိုင်ပါတယ်။ Python ဖိုင်တွေက .py အိုင်စာန်းရှင်းနှုပ်။ ကားရဲလ်ပရိုဂရမ်တစ်ခုကို Python ဖိုင်တစ်ခု ထားပါမယ်။ ပင်မ ပရောဂျက်ဖိုဒ်အောက်မှာပဲ တိုက်ရိုက်ရှိရပါမယ်။

နောက်ပိုင်း အဆင့်မြင့်လာရင် ပရိုဂရမ်တစ်ခုအတွက် ပရောဂျက်တစ်ခု ထားနိုင်တယ်။ ကုဒ်ဖိုင်တွေ အပြင် ပရိုဂရမ်အတွက် လိုအပ်တဲ့ ရုပ်ပုံတော့၊ အခြားဖိုင်တွေ (config ဖိုင်၊ setting ဖိုင် စသည်ဖြင့်) လည်း ပါနိုင်တယ်။ ပင်မပရောဂျက် အောက်မှာပဲ ဖိုင်တွေက တိုက်ရိုက်ရှိဖို့လည်း မလိုတော့ဘူး။ ဆက်စပ်ရာ ဖိုင်တွေကို အမျိုးအစားအလိုက် ဖန်ရှင်အလိုက် ဖို့ဒါတွေခဲ့ပြီး စနစ်ကျ စီစဉ်ဖွဲ့စည်း ထားရမှာပါ။ ပရောဂျက်တစ်ခုမှာ ဖိုင်တွေကို စနစ်တကျ စုဖွဲ့ထားဖို့ အရေးကြီးပါတယ်။



ပုံ ၂/၁၀

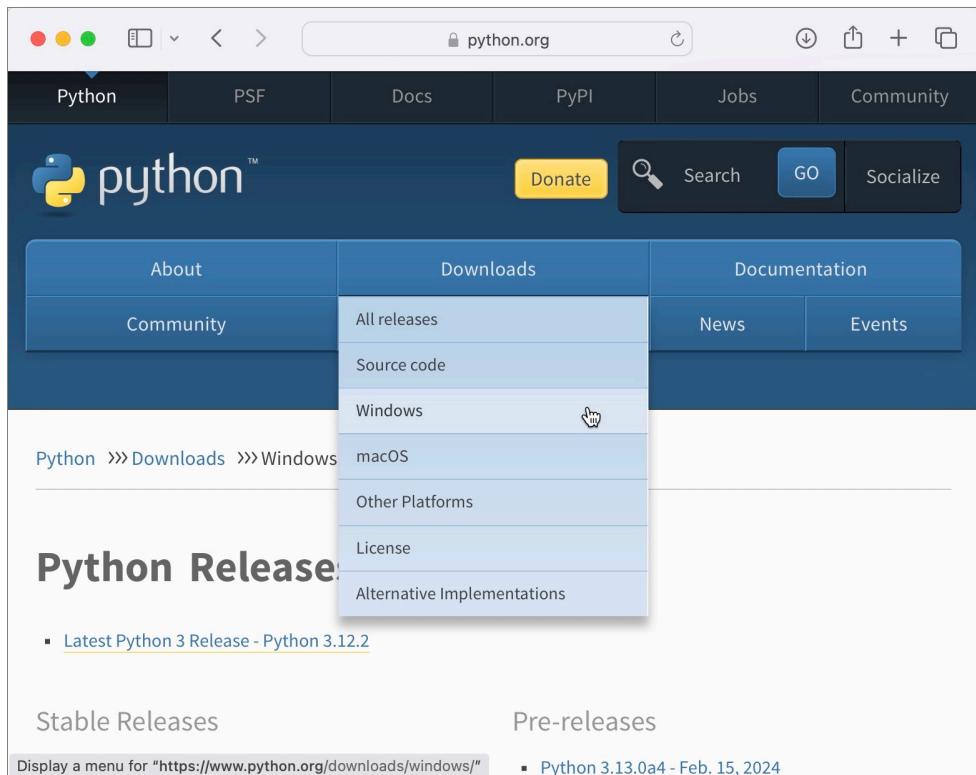
Visual Studio Code နှင့် Python အင်စတေလုပ်ခြင်း

Visual Studio Code(VS Code) ဟာ ပရိုဂရမ်မာအများစုံ ကြိုက်နှစ်သက်တဲ့ မောဒန် ကုဒ်အယ်ဒီတာ တစ်ခုပါ။ Python, JavaScript, C++ စတဲ့ programming language အမျိုးမျိုးအတွက် အသုံးပြု နိုင်ပါတယ်။

PyCharm မှာတော့ ပရောဂျက်အသစ်ဆောက်ရင် Python ပါ တစ်ခါတည်း ဒေါင်းလုဒ်လုပ်ပြီး အင်စတေလုပ်လိုလိုရတယ်။ VS Code နဲ့ Python ရေးမယ်ဆိုရင် Python Programming Language ကို သီး၌ ဒေါင်းလုဒ်လုပ်ပြီး အင်စတေလုပ်ရပါမယ်။

Python အင်စတေလုပ်လုပ်ခြင်း

ဒီလုပ် <https://www.python.org/> ကိုဖွေ့စ်ပါ။ Download မိန္ဒီးမှ Windows နိုပ်ပါ (ပုံ က/၁၁ ကို ကြည့်ပါ)။ Apple ကွန်ပျူးဘအတွက် ဆိုရင် macOS ရွေးရပါမယ်။ လူများစုသုံးတဲ့ ပိုက်ခရီးဆွေ့စွဲ ဝင်းခိုးအတွက် အင်စတေလုပ်နည်းကို အခိုက်ပြေားမှာပါ။ ပုံ က/၁၂ မှာလို တွေ့ရပါလိမ့်မယ်။ အင်စတော်လာ ဗားရှင်း 3.12 ထဲက လက်ရှိအမြင့်ဆုံး ကိုရွေးပါ။ ဒီစာရေးချိန်မှာ 3.12.2 ဟာ အမြင့်ဆုံးဗားရှင်းပါ။ Windows Installer (64-bit) ကိုနိုင်ပြီး ဒေါင်းလုဒ်လုပ်ပါ။



ပုံ က/၀၁

ဒေါင်းလုဒ်ပြီးရင် အင်စတော်လာဖိုင်ကို ညာကလစ်နိုင်ပြီး Run as administrator လုပ်ပါ။ ပုံ (က/၁၃) ကိုကြည့်ပါ။ ပုံ (က/၁၄) မှာလို ဒိုင်ယာလော် ဆောက်စ် ပွင့်လာပါမယ်။ အနိဂုံးထားတဲ့ ချက်ချေ ဆောက်စ်နှစ်ခုကို ချက်ချေလုပ်ပြီး Install Now နိုင်ပါ။ အင်စတေလုပ်ပြီးလို Setup was successful

Note that Python 3.12.2 cannot be used on Windows 7 or earlier.

- Download Windows embeddable package (32-bit)
- Download Windows embeddable package (64-bit)
- Download Windows embeddable package (ARM64)
- Download Windows installer (32-bit)
- Download Windows installer (64-bit)**
- Download Windows installer (ARM64)

Python 3.12.1 - Dec. 8, 2023

Note that Python 3.12.1 cannot be used on Windows 7 or earlier.

- Download Windows embeddable package (32-bit)
- Download Windows embeddable package (64-bit)
- Download Windows embeddable package (ARM64)
- Download Windows installer (32-bit)
- Download Windows installer (64-bit)

Python 3.13.0a2 - Nov. 21, 2023

- Download Windows embeddable package (32-bit)
- Download Windows embeddable package (64-bit)
- Download Windows embeddable package (ARM64)
- Download Windows installer (32-bit)
- Download Windows installer (64-bit)
- Download Windows installer (ARM64)

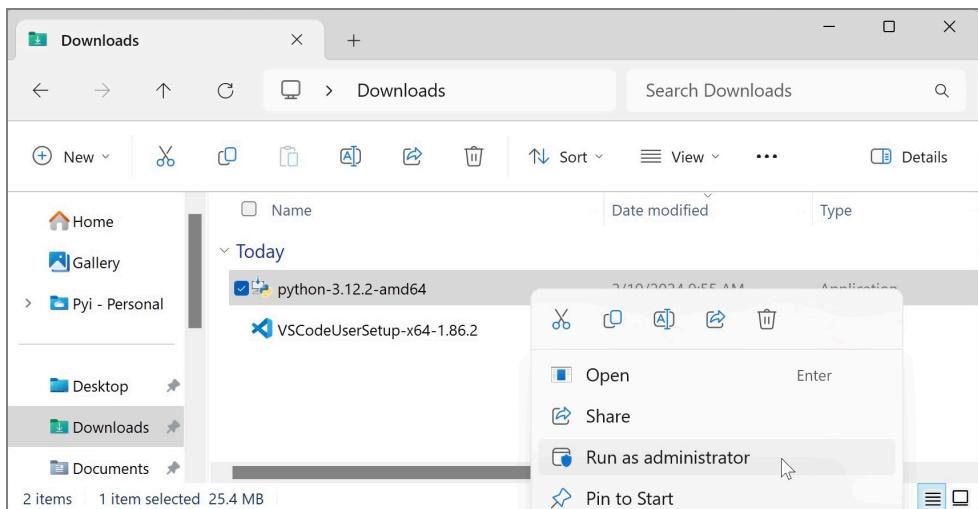
Python 3.13.0a1 - Oct. 13, 2023

- Download Windows embeddable package (32-bit)
- Download Windows embeddable package (64-bit)
- Download Windows embeddable package (ARM64)
- Download Windows installer (32-bit)
- Download Windows installer (64-bit)
- Download Windows installer (ARM64)

ပုံ ၂/၁၂

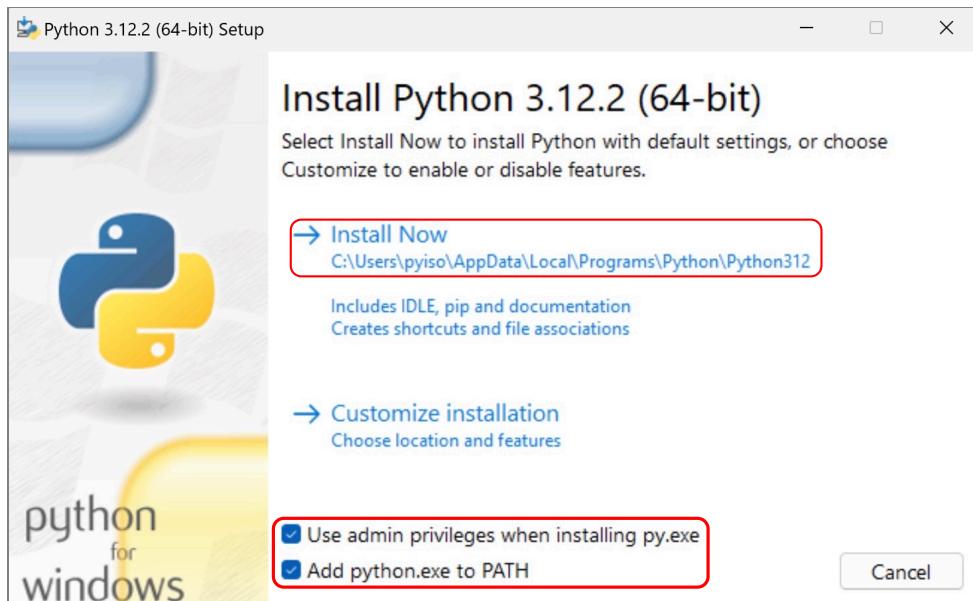
ပေါ်လာရင် Close ခလုတ်နိုပ်ပြီး ပိတ်ပါ။

ဝင်းဒီး command prompt (cmd) မှာ `python --version` run ရင် ပုံ (၃/၁၃) မှာလို အင် စကောလုပ်ထားတဲ့ ဗားရှင်းကို ပြပေးသင့်ပါတယ်။



ပုံ ၃/၁၃

፭



፭ ተ/ዕና

The screenshot shows a Windows Command Prompt window titled "Command Prompt". The text inside the window reads:

```
Microsoft Windows [Version 10.0.22621.2861]
(c) Microsoft Corporation. All rights reserved.

C:\Users\pyiso>python --version
Python 3.12.2

C:\Users\pyiso>
```

፭ ተ/ዕና

VS Code အင်စတေလုပ်ခြင်း

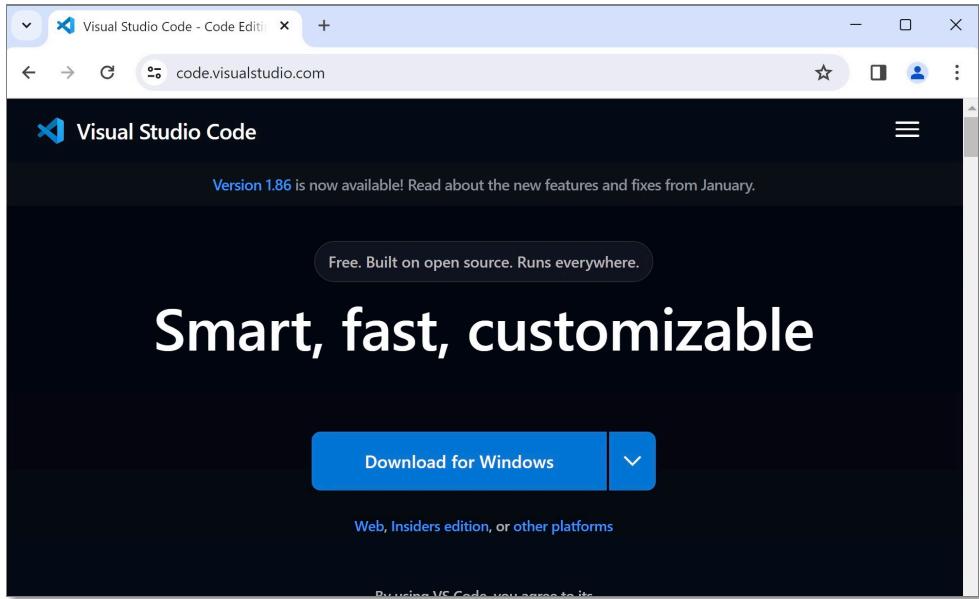
အခုနိရင် Python programming language အောင်မြင်စွာ ထည့်ပြီးသွားပါပြီ။ VS Code အင်စတေလုပ်ဆက်လုပ်ပါမယ်။ အင်စတေလူ ဒေါင်းလုပ်လုပ်ရန် ဝဘ်စာမျက်နှာကို အောက်ပါလုပ်လုပ်ပါ။

<https://code.visualstudio.com>

မှတစ်ဆင့် သွားပါ။ ပုံ (က/၁၆) ဝဘ်စာမျက်နှာကို တွေ့ရပါမယ်။ [Download for Windows](#) နှိပ်၍ ဒေါင်းလုပ်လုပ်ပါ။

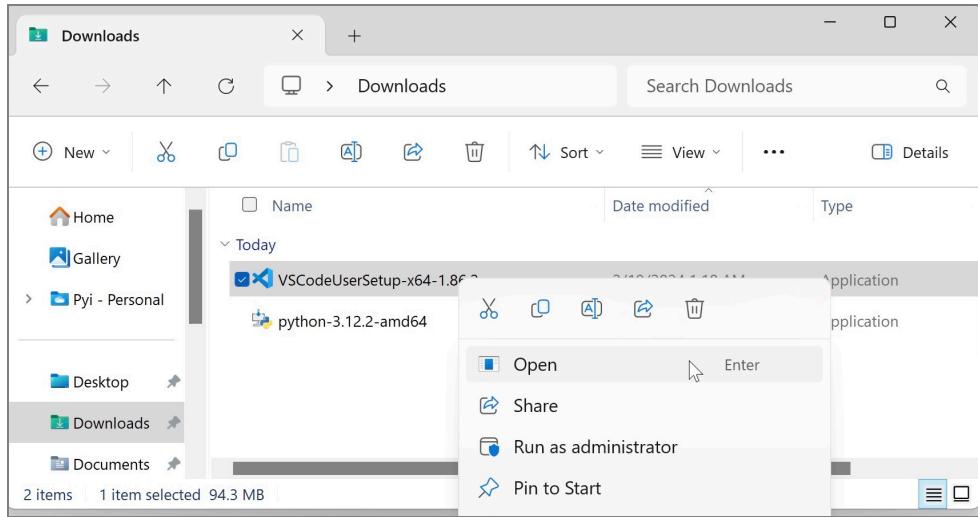
ပြီးတဲ့အခါ အင်စတေလုပ်ကို ညာကလစ်နိုင်ပြီး ဖွင့်ပါ (ပုံ က/၁၇ မှာ ပြထားပါတယ်)။ အင်စတေလုပ် ဒိုင်ယာလော်ဘောက်စ် ပွင့်လာရင် [I accept the agreement](#) ကို ချက်ချွဲလုပ်၍ Next > တစ်ခု ပြီးတစ်ခု ဆက်နှိပ်သွားပြီး နောက်ဆုံးမှာ Install နှိပ်ပါ။ အင်စတေလုပ်နေတာကို ခဏောင့်ပြီး၊ ပြီး သွားရင် Finish နှိပ်ပါ။

Welcome စခရင်ကို ပုံ (က/၁၈) လို တွေ့ရပါမယ်။ [Dark Modern](#) (သို့) [Light Modern](#) နှစ်သက်ရာ သို့မဟုတ် ရွေးပါ။ ဒါဆိုရင် VS Code လည်း အင်စတေလုပ်လုပ်ပါပြီ။ ကားရဲ့လုပ်ရုံးမှာ ရေးဖို့ ဆက်လုပ်ရပါမယ်။

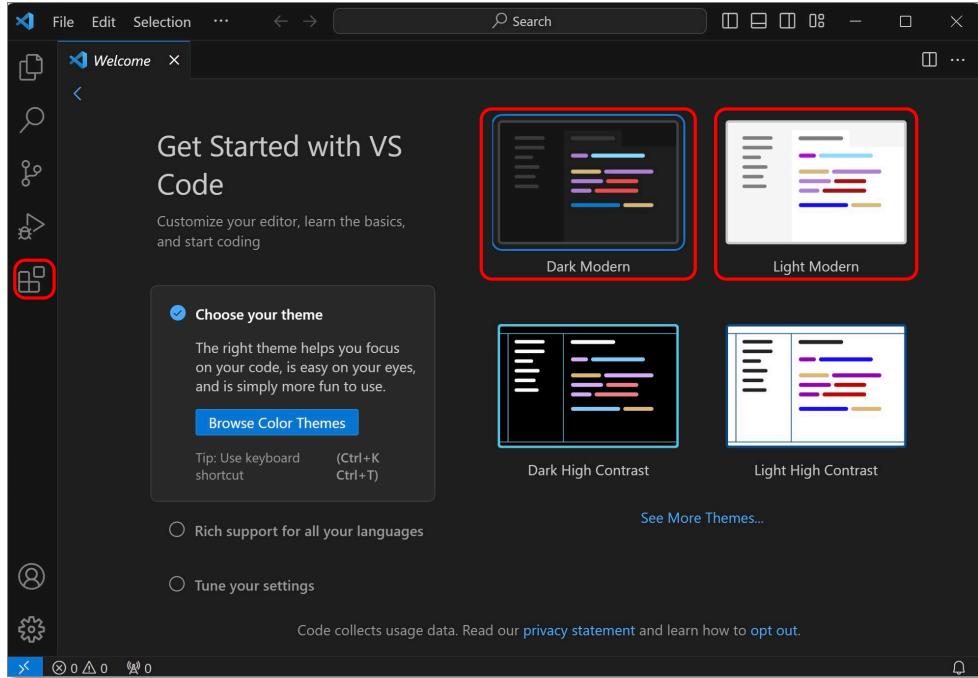


ပုံ က/၁၆

၁၃



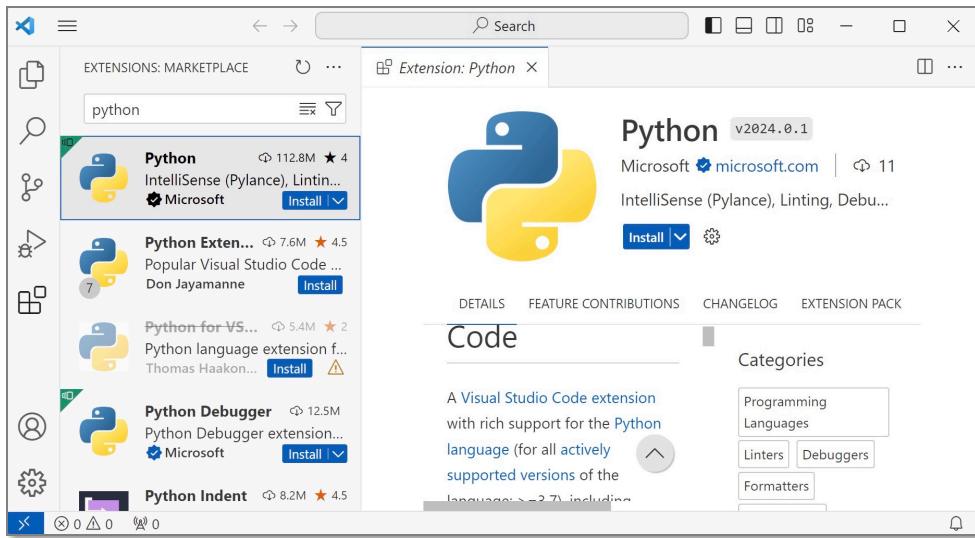
ပုံ ၂/၀၄



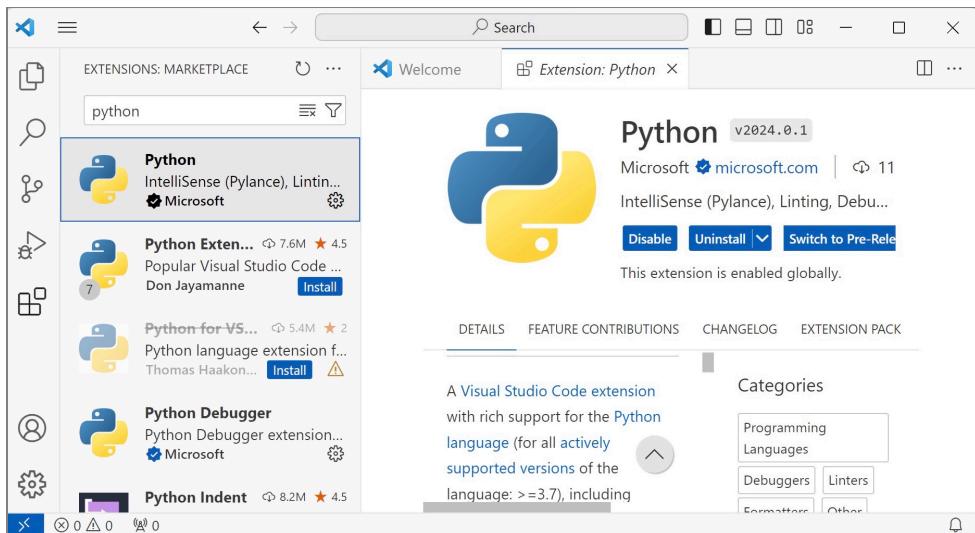
ပုံ ၃/၀၄

VS Code Python Extension တည်ခြင်း

Python ပရိဂုဂ္ဂများဖို့အတွက် VS Code က ဒီအတိုင်းဆိုရင် သိပ်အဆင်မပြေသေးပါဘူး။ Python အတွက် extension အင်စတောလ် လုပ်ပေးရပါအိုးမယ်။ ပုံ (က/၁၈) ဘယ်ဘက်ဘောင်နားမှာ အနီးပိုင်းထားတဲ့ အိုင်ကွန်လေးကို နှိပ်ပါ။ Python extension ကိုရှာပါ။ ပုံ (က/၁၉) မှာ ဘယ်လိုရှာရမလဲ ပြထားပါတယ်။ ပုံမှာတွေ့ရတဲ့ Microsoft က ထုတ်တဲ့ extension ကို အင်စတောလ်လုပ်ပါ။ အောင်မြင်ရင် ပုံ (က/၂၀) မှာလို ဖြစ်သွားပါမယ်။ Python နဲ့ VS Code ကိစ္စတော့ ပြီးသွားပြီ။ ကားရဲ့လဲ example run ဖို့ ဆက်လပ်ရမယ်။



ပုံ က/၁၉



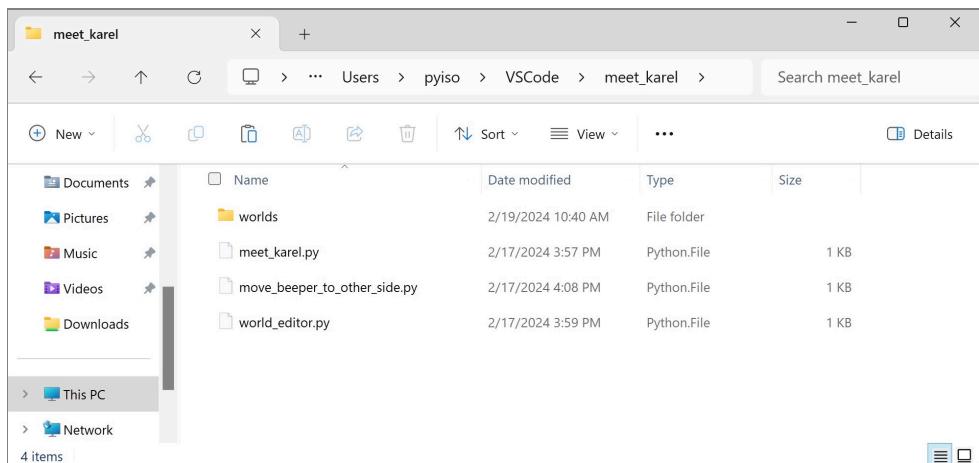
ပုံ က/၂၀

နမူနာ ကားရဲလ် ကဗ္ဗာနှင့် ပရိဂရမ်ဘ် ဖိုင်များထည့်ခြင်း

meet_karel.zip ဖိုင်ကို ဒီလင့် <http://tinyurl.com/3mm9c7j> ကနေ ဒေါင်းလုပ်လုပ်ပါ။ ငှါး zip ဖိုင်ကို extract လုပ်ပါ။ meet_karel နံမည်နဲ့ ဖိုဒါတစ်ခု ရလာပါမယ်။ ဖိုဒါထဲ ဝင်ကြည့်ရင် အောက်ပါ အတိုင်း ရှိသင့်ပါတယ်။

■ worlds

- ▶ meet_karel.w
- ▶ move_beeper_to_other_side.w
- meet_karel.py
- move_beeper_to_other_side.py
- world_editor.py



ပုံ က/ဂ

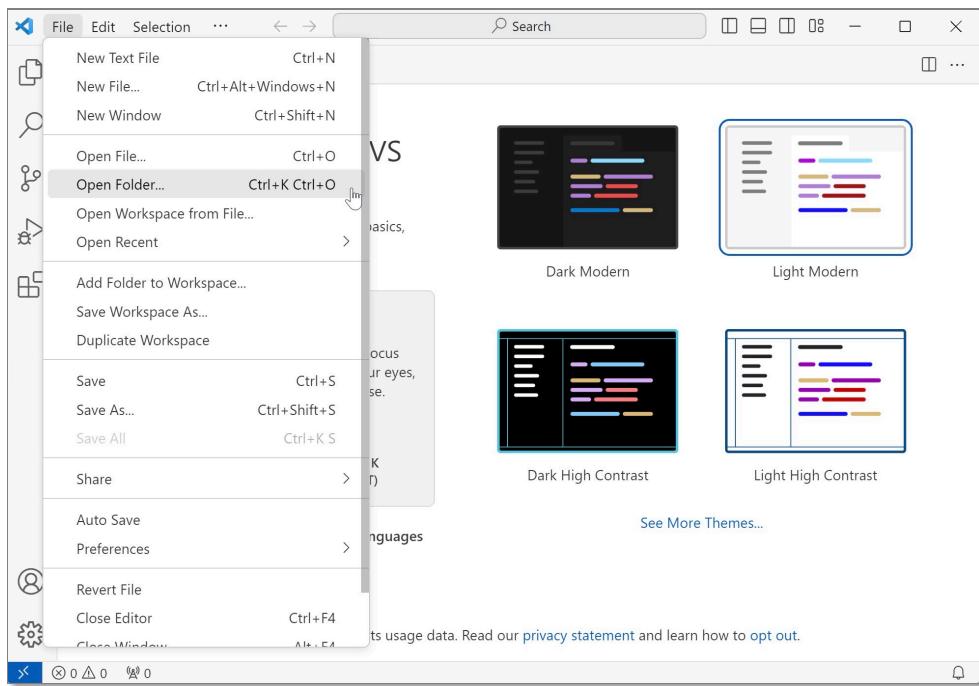
VS Code အတွက် ဖိုဒါတစ်ခုကို မိမိအတွက် အဆင်ပြေမဲ့နေရာမှာ သီးသန့်တည်ဆောက် ထားသင့်တယ်။ ဥပမာ

C:\Users\yourname\VS Code

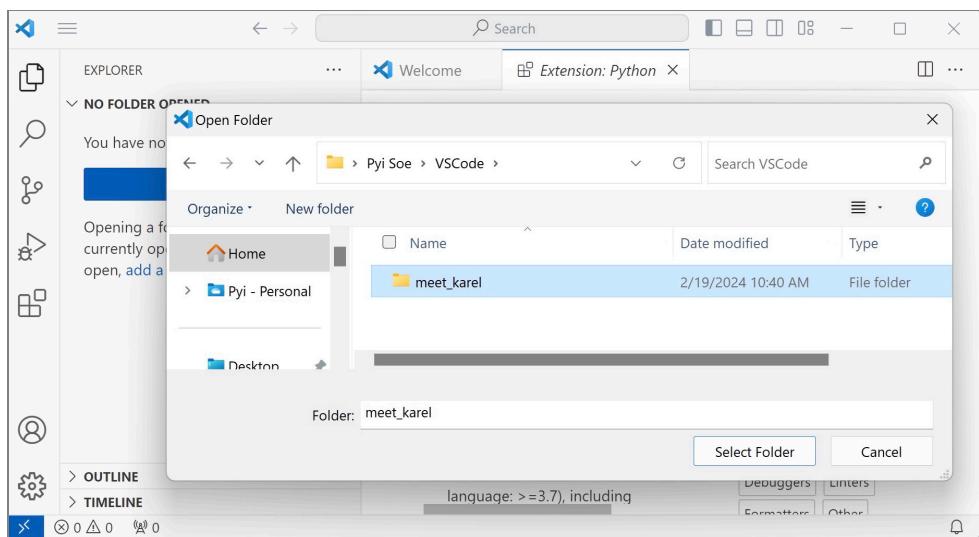
မိမိ လက်ရှိ Home ဖိုဒါကို C: drive ရဲ့ Users ဖိုဒါထဲမှာ တွေ့နိုင်ပါတယ်။ Win + R ရှေ့ကတ်နှုပ်ပြီး %userprofile% ရှိက်ထည့်။ Ok လုပ်ပြီး Home ဖိုဒါကို သွားနိုင်ပါတယ်။ အကယ်၍ မသွားတတ်ရင်လည်း ပြဿနာမရှိပါဘူး။ ကိုယ့်အတွက် လွယ်ကူမဲ့ Desktop, Downloads, Documents တစ်ခုစုထဲမှာ VS Code အတွက် ဖိုဒါတစ်ခု ထားလည်းရတယ်။

`meet_karel` ဖိုဒါ (.zip ဖိုင်မဟုတ်ပါ) ကို အထက်ပါအတိုင်း အသစ် ဆောက်ထားတဲ့ VS Code သီးသန့်ဖိုဒါထဲကို ကော်ပီကူးထည့်ပါ။ ငှါး `meet_karel` ဖိုဒါကို VS Code `File` မိုးမှ `Open Folder` နှုပ်၍ ဖွေ့စွဲပါ။ ပုံ (က/ဂ) တွင်ကြည့်ပါ။

အခန်းအလိုက် နမူနာ ကုဒ်ဖိုင်တွေ ထည့်ပေးထားတဲ့ .zip ဖိုင်တွေကိုလည်း အထက်ပါအတိုင်း လုပ်ရပါမယ်။ .zip ဖိုင်ကို ဖြည့်၊ ရလာတဲ့ ဖိုဒါကို သီးသန့်ဖိုဒါတစ်ခုထဲကို ကော်ပီကူးထည့်၊ VS Code နဲ့ အဲ ဒီဖိုဒါကို ဖွေ့စွဲပါပဲ။



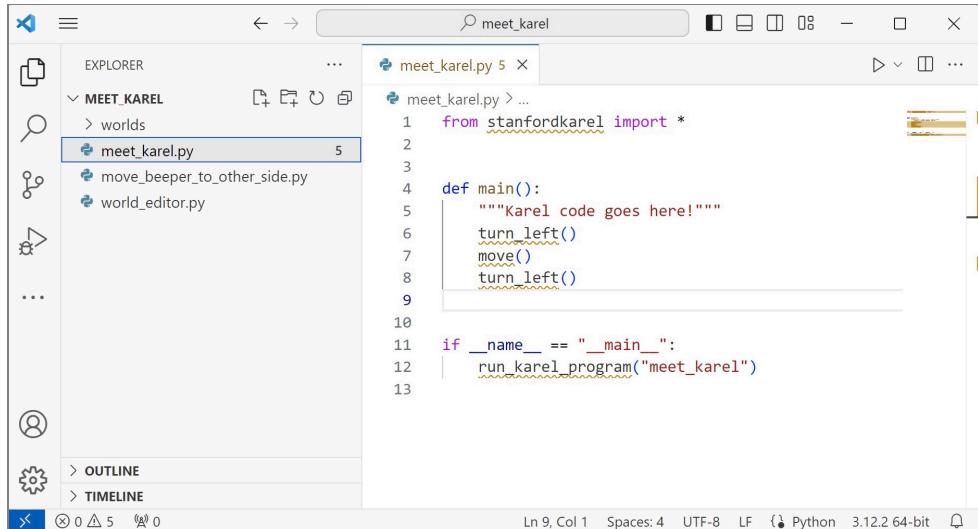
ပုံ က/JJ



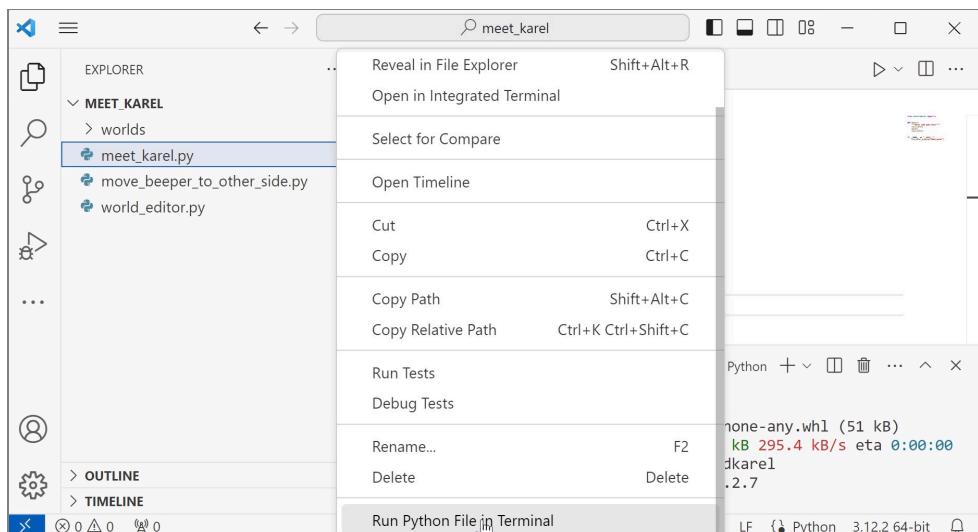
ပုံ က/JR

stanfordkarel လိုက်ဘရီ အင်စတောလ်လုပ်ခြင်း

`meet_karel.py` ဖိုင်ကို ကလစနှစ်ချက်နှင့် ဖွင့်ပါ။ ကုဒ်အယ်ဒီတာ ပွင့်လာမယ် (ပုံ ၂/၂၅)။ အဲဒီ ကုဒ် အယ်ဒီတာပေါ် (သို့) `meet_karel.py` ဖိုင်ကို ညာကလစနှစ်ပြီး `Run Python File in Terminal` လုပ်ပါ (ပုံ ၂/၂၅)။ Terminal ပွင့်လာပြီး အယ်ရာမက်ဆောင်တွေ ပြလိမ့်မယ်။ ပုံ (၂/၂၆) မှာ ကြည့်ပါ။ ကဗျားရုံးပရှုဂျုပ်အတွက် လိုအပ်တဲ့ stanfordkarel လိုက်ဘရီ အင်စတောလ် မလုပ်ရသေးပါဘူး။ ဒါ ကြောင့် အယ်ရာဖြစ်နေတာ။



ပုံ ၂/၂၆



ပုံ ၂/၂၇

ခုနကပွင့်လာတဲ့ Terminal မှာပဲ အောက်ပါကွန်မန်းကို run ပြီး stanfordkarel လိုက်ဘရီကို အင်စတောလ်လုပ်ပါ။

```

from stanfordkarel import *
def main():
    """Karel code goes here!"""
    move()
    move()
    move()
    pick_beeper()

```

PROBLEMS 5 OUTPUT TERMINAL ...

PS C:\Users\pyiso\VSCode\meet_karel> & C:/Users/pyiso/AppData/Local/Programs/Python/Python312/python.exe c:/Users/pyiso/VSCode/meet_karel/meet_karel.py
Traceback (most recent call last):
 File "c:/Users/pyiso/VSCode/meet_karel/meet_karel.py", line 1,
 in <module>
 from stanfordkarel import *
ModuleNotFoundError: No module named 'stanfordkarel'
PS C:\Users\pyiso\VSCode\meet_karel> pip install stanfordkarel

Ln 9, Col 1 Spaces: 4 UTF-8 LF { Python 3.12.2 64-bit }

ပုံ ၃/၂၆

`pip install stanfordkarel`

ပုံ (၃/၂၆) မှာ အနိဂင်းထားတာကို ကြည့်ပါ။ အဲဒီအတိုင်းရိုက်ထည့်ပြီး Enter ကိုနိပ်ပါ။ ခဏကြောတဲ့ အခါ အခုလို မက်ဆော်ချုပ်တွေ ကျလာပါလိမ့်မယ်။

```

ModuleNotFoundError: No module named 'stanfordkarel'
PS C:\Users\pyiso\VSCode\meet_karel> pip install stanfordkarel
Collecting stanfordkarel
  Downloading stanfordkarel-0.2.7-py3-none-any.whl (51 kB)
           ━━━━━━━━━━━━ 51.9/51.9 kB 295.4 kB/s eta 0:00:00
Installing collected packages: stanfordkarel
Successfully installed stanfordkarel-0.2.7
PS C:\Users\pyiso\VSCode\meet_karel>

```

ဟိုကလိုက်ပြထားတဲ့ မက်ဆော်ချုပ် တွေ့ရင် အင်စတောလ် အောင်မြင်လိုပါ။ ပုံ (၃/၂၇) အယ်ဒီတာမှာလို သတိပေး လိုင်းတွေ့နှုန်းတွေ မရှိသင့်တော့ဘူး။ stanfordkarel လိုက်ဘရဲ့ အင်စတောလ် လုပ်ပြီးပြီးလို့ သတိပေးတာတွေ ပျောက်သွားသင့်တယ်။

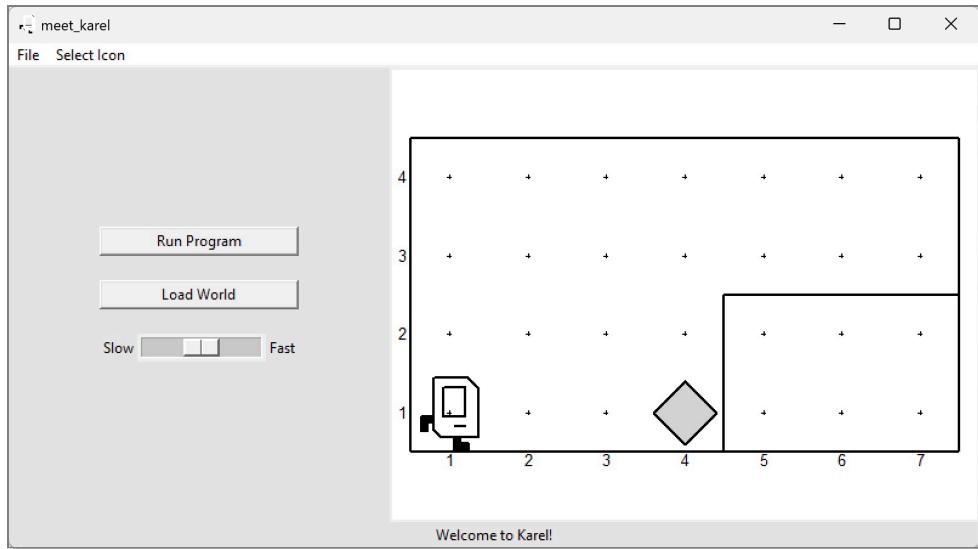
meet_karel.py ဖိုင်ကို ညာကလစ်နှုပ်ပြီး Run Python File in Terminal ပြန်လုပ်ကြည့်ပါ။ ပုံ (၃/၂၇) မှာပြထားတဲ့ ကားရဲလ်ပရိုကရမ် ပွင့်လာသင့်ပါတယ်။ Run Program နှုပ်ကြည့်ပါ။ စက်ရှုပ် လေး ကားရဲလ် နေရာရွှေသွားတာ တွေ့ရမယ်။ meet_karel.py ကို အောက်ပါအတိုင်း ဖြည့်စက်ရေးပါ။

```

from stanfordkarel import *

def main():
    """Karel code goes here!"""
    move()
    move()
    move()
    pick_beeper()

```



ဗုံး က/ဂျ

```

turn_left()
move()
move()

turn_left()
turn_left()
turn_left()

move()
put_beeper()

if __name__ == "__main__":
    run_karel_program("meet_karel")

```

`meet_karel.py` ဖိုင်ကို ညာကလစ်နိုပ်ပြီး Run Python File in Terminal ပြန်လုပ်ကြည့်ပါ။ ကား ရဲလ်ပရိုကရမ် ပွင့်လာရင် Run Program နှိပ်ကြည့်ပါ။ ဘိပါလေးကို နေရာရွှေ့ပေးပါလိမ့်မယ်။ အကယ်၍ ကားရဲလ်ပရိုကရမ် မတက်လာရင် ကုဒ်ရေးတာမှားနေလို့ ဖြစ်နိုင်တယ်။ အပေါ်ကုဒ်နဲ့ နှိုင်းယှဉ်ပြီး ကြည့်ပါ။ VS Code အယ်ဒီတာမှာ အနိုင်းတွေ့လေးတွေ ပြတဲ့နေရင် အဲဒီနေရာတွေမှာ ဆင်းတက်စုံမှားနေတာ ဖြစ်နိုင်တယ်။

VS Code အယ်ဒီတာမှာ ပရိုကရမ်ကုဒ်ပြင်ပြီး ပြန် run တဲ့အခါ ပထမ run ထားတဲ့ ပရိုကရမ်ကို အ ရင်ပိတ်ဖို့လိုပါတယ်။ ခိုလိုတာက `meet_karel.py` ကို run ထားတယ်ဆိုပါစိုး။ ပုံ (က/ဂျ) က ဝင်း ဒိုးပွင့်လာမယ်။ `meet_karel.py` ကုဒ်ကို ပြင်ပြီး ပြန် run ချင်ရင် အဲဒီ ဝင်းဒိုးကို အရင်ပိတ်ရမယ်။ မဟုတ်ရင် ပြင်ထားတဲ့ ပရိုကရမ်က ချက်ချင်း ပွင့်မလာဘူး။ ပထမ ဟာကို ပိတ်တော့မှုပဲ နောက် run တဲ့ဟာ ပွင့်လာမှာ။

ဝိုက်ကွင်းကျန်နေတာက အဖြစ်များတဲ့ အမှားပါ။ ကျန်ခဲ့လို့ မရပါဘူး။ အင်ဒန်တေးရှင်း (indentation)

tion) လုပ်ရမဲ့နေရာမှာ မလုပ်ထားရင်လည်း ပြဿနာဖြစ်တယ်။ move, turn_left တွေကို ဘေးမျဉ်းလှာဘက်ဆွဲပြီး အင်ဒန်လုပ်ပေးရမယ်။ အဒါတွေ ဂရာမစိုက်မိရင် ဆင်းတက်စံအမှားဖြစ်ပြီး ပရိုဂရမ် run လို့ မရနိုင်ဘူး။

Terminal မှာ ထုတ်ပေးတဲ့ မက်ဆွဲချုပ်တွေကို ကြည့်ပြီးတော့လည်း ဘာပြဿနာဖြစ်နေလဲ မှန်းဆလို့ရနိုင်တယ်။ ဘာကြောင့်ဖြစ်နိုင်လဲ ဆက်စပ်စဉ်းစားလို့ ရတယ်။ ဥပမာ ဖြစ်တဲ့ပြဿနာအလိုက် အခုလို တွေ့ရပါမယ်။

```
File "c:\Users\pyiso\VS Code\meet_karel\meet_karel.py", line 6
    move(
        ^
SyntaxError: '(' was never closed

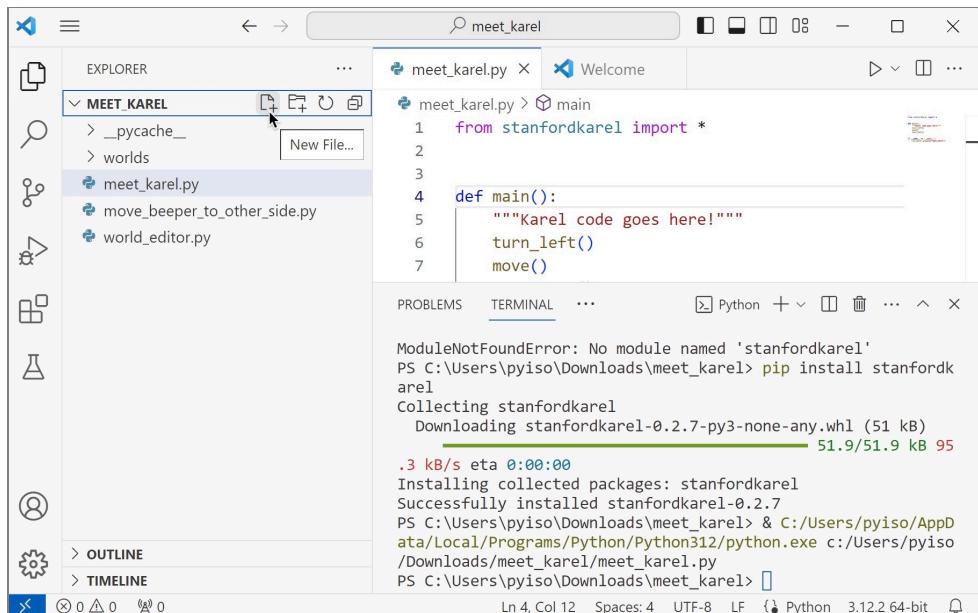
File "c:\Users\pyiso\VS Code\meet_karel\meet_karel.py", line 7
    move()
        ^
IndentationError: unexpected indent

Traceback (most recent call last):
  File "c:\Users\pyiso\VS Code\meet_karel\meet_karel.py", line 1,
    in <module>
      from stanfordkarel import *
ModuleNotFoundError: No module named 'stanfordkarel'
```

VS Code တွင် Python ဖိုင် အသစ်ယူခြင်း

MEET_KAREL ပင်မ ပရောဂျက်ဖိုဒ်အပေါ်မှာ ကလစ်နိုင်ပါ။ ပုံမှာ ပြထားတဲ့ New File အင်ကွန်ကိုယူပိုင်ပါ။ ဖိုင်နံပည်ဖြည့်တဲ့ ဘောက်စံလေး ပေါ်လာမယ်။ Python ဖိုင်တွေက .py အိပ်စံတန်းရှင်းနဲ့ ဖြစ်ရပါမယ်။ ဒါကြောင့် နံပည် ဖြည့်တဲ့အခါ .py နဲ့ အဆုံးသတ်ပေးရပါမယ် (ဥပမာ hello.py)။ ကားချုပ်ပရီဂရမ်တစ်ခုကို Python ဖိုင်တစ်ခု ထားပါမယ်။ ပင်မ ပရောဂျက်ဖိုဒ်အပေါ်မှာပဲ တိုက်ရှိကြရပါမယ်။

နောက်ပိုင်း၊ အဆင့်မြင့်လာရင် ပရီဂရမ်တစ်ခုအတွက် ပရောဂျက်တစ်ခု ထားနိုင်တယ်။ ကုဒ်ဖိုင်တွေ အပြင် ပရီဂရမ်အတွက် လိုအပ်တဲ့ ရုပ်ပုံတော့၊ အခြားဖိုင်တွေ (config ဖိုင်၊ setting ဖိုင် စသည်ဖြင့်) လည်း ပါနိုင်တယ်။ ပင်မပရောဂျက် အောက်မှာပဲ ဖိုင်တွေက တိုက်ရှိကြရဖို့လည်း မလိုတော့ဘူး။ ဆက်စပ်ရာ ဖိုင်တွေကို အမျိုးအစားအလိုက်၊ ဖန်ရှင်အလိုက် ဖို့ဒါတွေခဲ့ခြား စနစ်ကျ စီစဉ်ဖွဲ့စည်း ထားရမှာပါ။ ပရောဂျက်တစ်ခုမှာ ဖိုင်တွေကို စနစ်တကျ စွဲထားဖို့ အရေးကြီးပါတယ်။



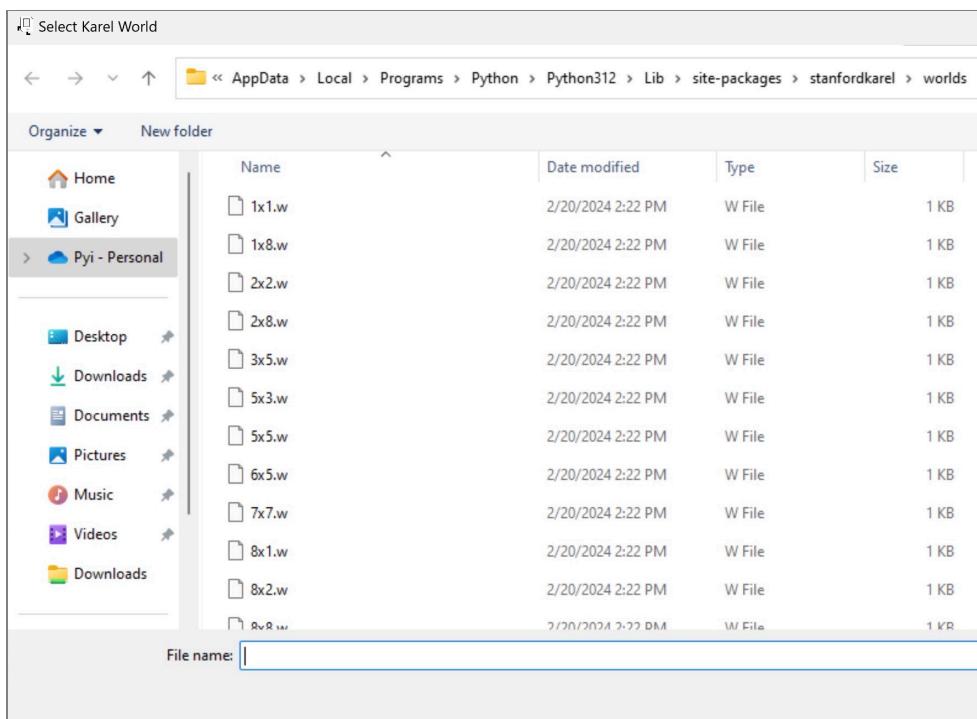
ပုံ က/၂

နောက်ဆက်တဲ့ ခ

ကားရဲလ်ပရီဂရမ် ဖီချာများ

ကားရဲလ် ကဗ္ဗာဖိုင်များ

ကားရဲလ်ပရီဂရမ် ဝင်းခွဲးမှာ **Load World** ခလုတ်နှုပ်ပြီး ကဗ္ဗာဖိုင်အသစ် တင်လိုရတယ်။ ခလုတ်နှုပ်လိုက် ရင် အခုလို ဖိုင် ခိုင်ယာလော့ဂုံးပွင့်လာမယ်။



ပုံ ၉/၁

ဒါက stanfordkarel လိုက်ဘရဲ သူနိုင်အရို့အတိုင်း ပါတဲ့ worlds ဖို့ပါ။ ဖိုင်တွေက .w နဲ့ ဆုံးပါတယ်။ ကဗ္ဗာဖိုင်တွေကို ပင်မ ပရောဂျက်အောက် worlds ဖို့ပါထဲမှာ ထားလိုလည်းရတယ်။ အခြားနေရာတွေမှာ ထားလိုတော့ မရဘူး။

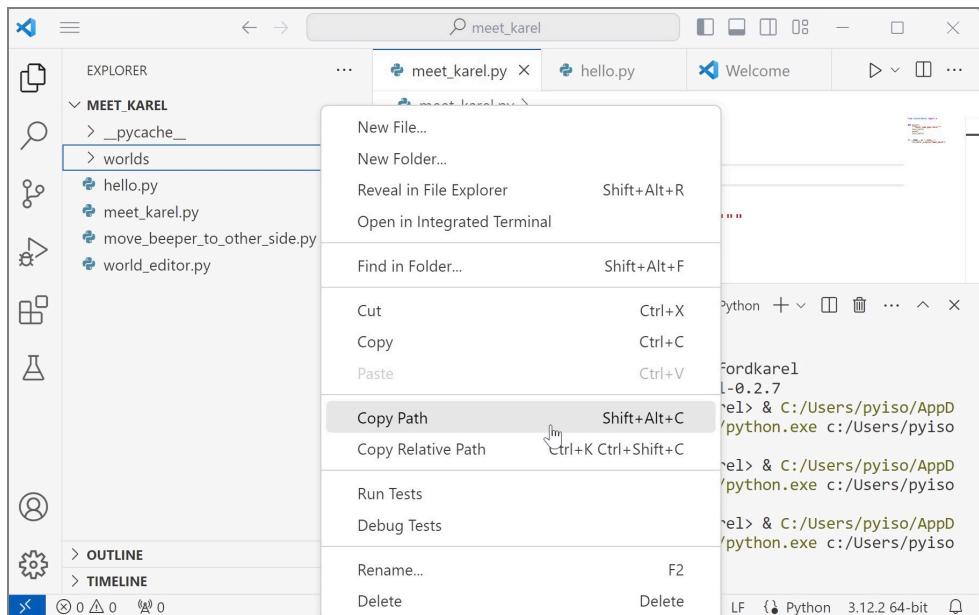
စာအုပ်ပါ ဥပမာတွေ၊ လေ့ကျင့်ခန်းတွေအတွက် ကမ္ဘာဖိုင်တွေကို ပရောဂျက် တစ်ခုချင်းအလိုက် သီးခြား worlds ဖိုဒါထဲမှာ ထည့်ပေးထားမှာပါ။ ပရိုဂ်ရမ်တစ်ခုဟာ ကမ္ဘာတစ်ခုတည်းမှာပဲ အလုပ်လုပ်တာမဟုတ်ဘဲ အလားတူ ကမ္ဘာအမျိုးမျိုးအတွက် အလုပ်လုပ်အောင် ရေးပေးရတာ။ အခြားတာကို နားမလည်ရင် အခန်း (၂) ဖတ်ပြီးရင် နားလည်သွားမှာပါ။

Load World လုပ်တဲ့အခါ ပွင့်လာတဲ့ ဖိုင် ခိုင်ယာလော်ကာ ကိုယ်လိုချင်တဲ့ worlds ဖိုဒါ မဖြစ်နေဘူး။ သူနိုင်ပါတဲ့ worlds ဖိုဒါ ဖြစ်နေတယ်။ ကိုယ်ခေါ်တင်ချင်တဲ့ ဖိုင်တွေရှိတဲ့ လက်ရှိပရောဂျက်ရဲ့ worlds ဖိုဒါကို သွားရမယ်။ ဥပမာ PyCharm/VS Code အတွက် MeetKarel/meet_karel ပရောဂျက် worlds ဖိုဒါ လမ်းကြောင်း အပြည့်အစုံက

```
C:\Users\yourname\VSCode\meet_karel\worlds
C:\Users\yourname\PycharmProjects\MeetKarel\worlds
```

ဖြစ်ပါမယ်။ ဖိုင်ခိုင်ယာလော်ကနေ ဖော်ပြပါ လက်ရှိပရောဂျက် worlds ဖိုဒါကို တစ်ဆင့်ချင်း သွားပြီး တင်ချင်တဲ့ ကမ္ဘာဖိုင် (.w ဖိုင်) ကို ရွေးရမှာပါ။

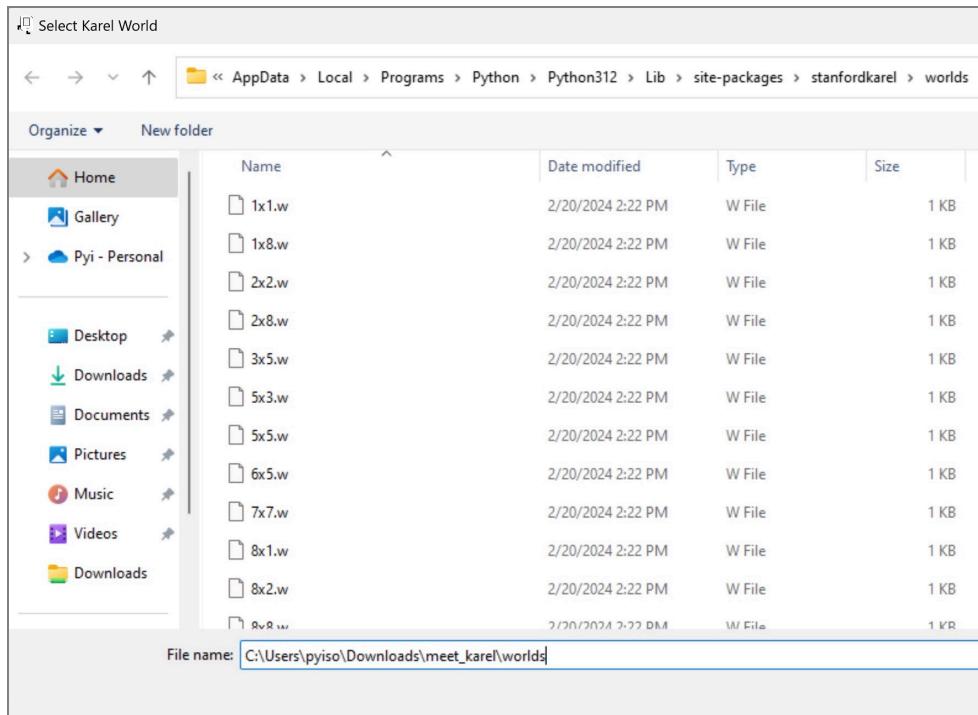
အပေါ်ကနည်းနဲ့ အဆင်မပြောရင် အခုလုံစမ်းကြည့်ပါ။ လက်ရှိပရောဂျက် worlds ဖိုဒါကို ညာကလစ်နှုပ်ပြီး Copy Path လုပ်ပါ (ပုံ ၁/၂)။ ဖိုင် ခိုင်ယာလော် **File name** မှာ ကူးထည့်ပါ (ပုံ ၂/၃)။ **Enter** ကိုနှိပ်ပါ။ ပရောဂျက် worlds ဖိုဒါကိုရောက်သွားပါမယ်။ လိုတဲ့ကမ္ဘာဖိုင် ရွေးတင်ရုံပါပဲ။ ပုံ (၁/၃) မှာ meet_karel worlds ကို နူးနားပြထားပါတယ်။



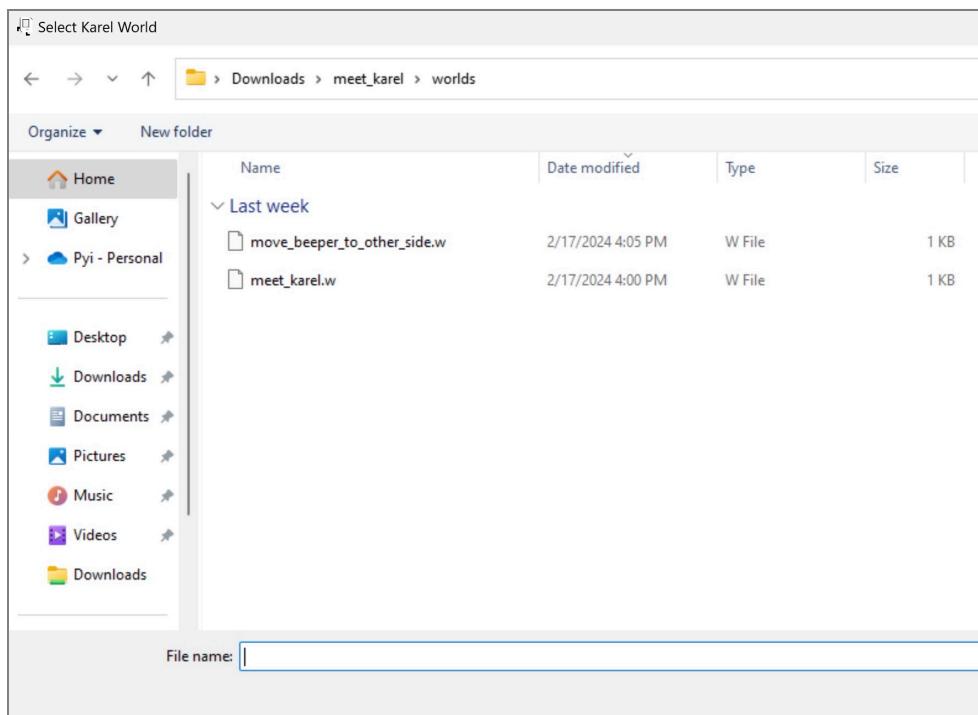
ပုံ ၁/၂

အကယ်၍ အထက်ဖော်ပြပါနည်းတွေက ရှုပ်နေတယ်ထင်ရင် မှတ်ရ/သွားရ လွယ်တဲ့ Desktop, Downloads, Documents လို နေရာတစ်ခုခုမှာ သီးသန့်ဖိုဒါတစ်ခု အောက်ပြီး ပရောဂျက်အားလုံး ထည့်ထားတာ အရှင်းဆုံးပါပဲ။ ပရောဂျက်ဖိုဒါနေရာ သိရင် ပိုင်ခိုင်ယာလော်ကနေ ဘယ်လိုမဆိုရောက်အောင် သွားလိုရပါတယ်။

၁၁



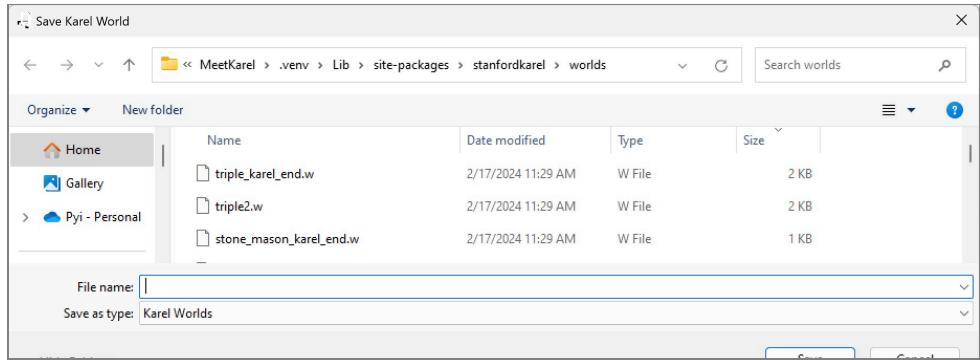
ပုံ ၅/၃



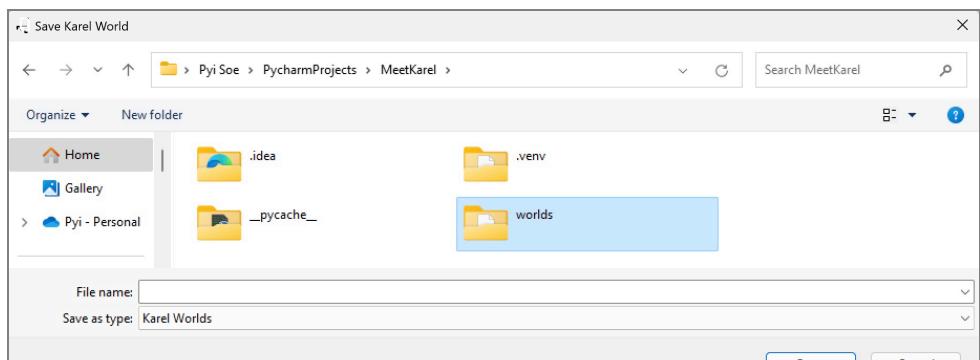
ပုံ ၆/၄

ကိုယ်ပိုင် ကားရဲလ်ကမ္ဘာ ဆောက်ခြင်း

ကားရဲလ်ရဲ ကမ္ဘာ အသစ်တစ်ခုဆောက်မယဆိုရင် world_editor.py ဖိုင်ကို ညာကလစ်နှင့် Run ပါ။ Would you like to load an existing world? လို ပေါ်လာပြီး Yes/No ရွှေ့ခိုင်ပါလိမ့်မယ။ No ကို နှိပ်ပါ။ ကမ္ဘာအရွယ်အစားအတွက် ကော်လံ ဘယ်နှစ်ခုလဲ row ဘယ်နှစ်ခုလဲ ထည့်ပေးပါ။ ကိုယ်ပိုင် ကားရဲလ်ကမ္ဘာ တည်ဆောက်လိုရတဲ့ ဝင်းဒီးပွင့်လာပါလိမ့်မယ။ ကားရဲလ် မျက်နှာမူရာအရပ်၊ ဘိပါအိတ်ထဲရှိ ဘိပါအရေအတွက်၊ နံရုံဆောက်/ဖျက်တာ၊ ဘိပါထည့်/ဖယ်ထုတ်တာ စတာတွေ လုပ်နိုင်ပါတယ်။ Save World နှိပ်ပြီး သိမ်းနိုင်ပါတယ် ဖိုင်ကိုသိမ်းတဲ့အခါ သုန္တရှိ သိမ်းခိုင်းတဲ့ဖို့ (default world folder) ထဲမှာ သို့မဟုတ် ပင်မ ပရောဂျက်ဖို့တဲ့က worlds ဖို့ဒါထဲမှာ သိမ်းရပါမယ်။



ပုံ ၉/၅



ပုံ ၉/၆

Python programming language

import, from, def, if

New term

fEnEmpstatement

ပါရာမီတာမပါရင်လည်း စိုက်ကွင်းအလွတ် တစ်ဖုံး ‘()’ တော့ပါရမယ်။

colon ‘:’

ဖန်ရှင်နဲ့ သက်ဆိုင်တဲ့ ကုဒ်ဘလောက် (*code block*)

quote သုံးခုတွဲ ‘“”’

ပရိုဂရမ်ဝင်းဒီးမှာ **Run Program** ခလုတ်

meet_karel.w ကမ္မာကို