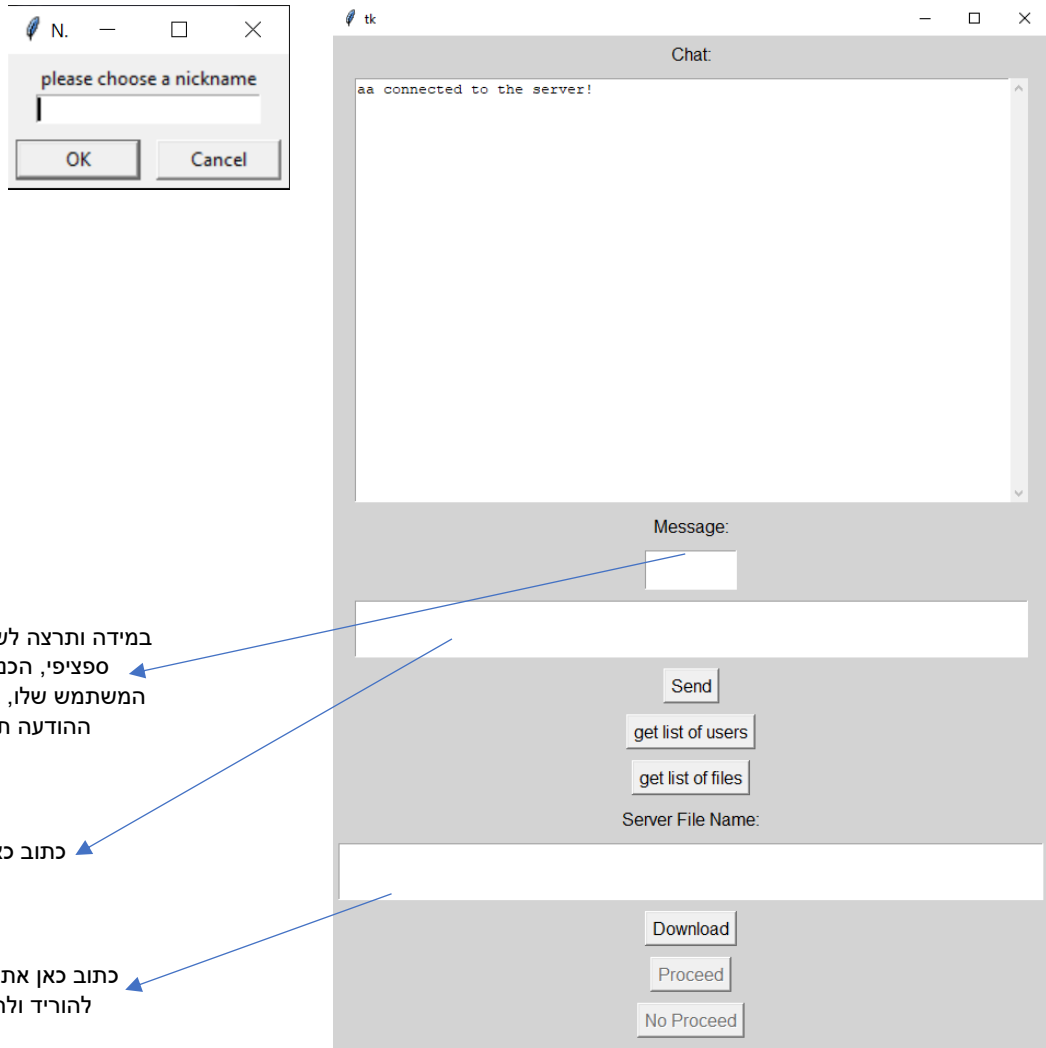


איך להריץ?

- הרץ את server
- הרץ את client_gui
- ייפתחו שני חלונות, בחלון הקטן הזן את השם משתמש ולחץ ok
- אתה מחובר ויכול לשוחח בצ'אט!



שים לב!

- על מנת להריץ את הטסטים יש להריץ קודם כל את השרת!

- יש לוודא שהספרייה numpcy מותקנת

UML

my_network.MyServer_TCP

```
m __init__(self, host='localhost', port=50000)
m broadcast_message(self, clients, message)
m get_host(self)
m wait_for_client_connection(self)
m send_UDP_port_to_client(self, client, message)
m send_message_to_client(self, client, message)
m get_message_from_client(self, client)
m wait_for_client_connection_and_get_message_from_client(self)
m disconnect_from_client(self, client)
m __create_server(self)
m __string_to_bytes(self, message_as_string)
m __bytes_to_string(self, message_as_bytes)
f __host
f __server
f __port
f __INCOMING_MESSAGE_BUFFER_SIZE
f __ENCODING
```

my_network.MyServer_UDP

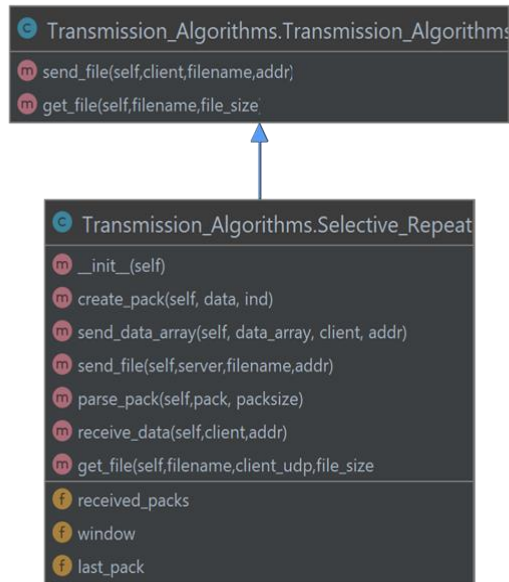
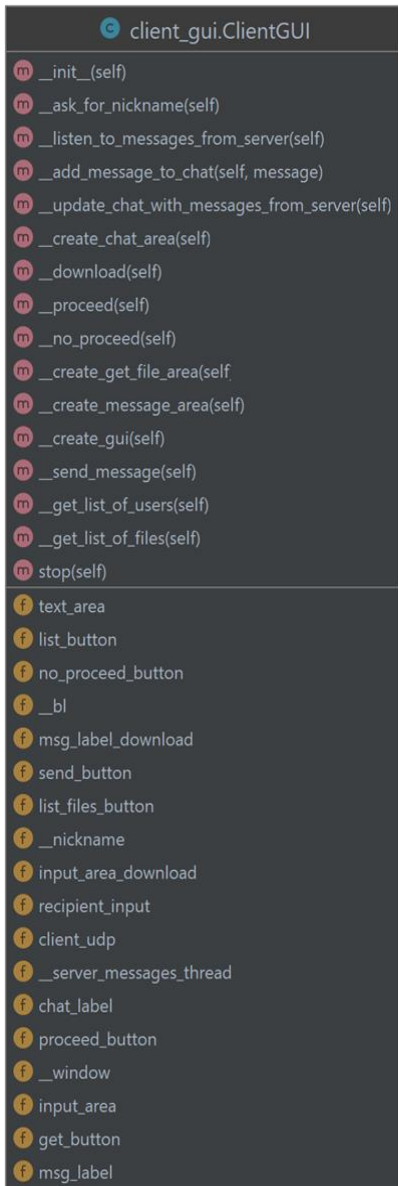
```
m __init__(self, host='localhost', port=19090)
m broadcast_message(self, clients, message)
m send_message_to_client(self, client, message)
m get_message_from_client(self, client)
m wait_for_client_connection_and_get_message_from_client(self)
m disconnect_from_client(self, client)
m get_sock(self)
m __create_server(self)
m __bind(self)
m __string_to_bytes(self, message_as_string)
m __bytes_to_string(self, message_as_bytes)
f __host
f __server
f __port
f __INCOMING_MESSAGE_BUFFER_SIZE
f __ENCODING
```

my_network.MyClient_UDP
















```
m __init__(self, server_host, server_port)
m connect(self)
m bind(self, UDP_IP, UDP_PORT)
m disconnect(self)
m send_message(self, message)
m get_message_from_server(self)
m __string_to_bytes(self, message_as_string)
m __bytes_to_string(self, message_as_bytes)
m get_sock(self)
m get_host(self)
m get_port(self)
f __sock
f __server_host
f __server_port
f __INCOMING_MESSAGE_BUFFER_SIZE
f __ENCODING
```








my_network.MyClient_TCP

```
m __init__(self, server_host, server_port)
m connect(self)
m disconnect(self)
m send_message(self, message)
m get_message_from_server(self)
m __string_to_bytes(self, message_as_string)
m __bytes_to_string(self, message_as_bytes)
f __sock
f __server_host
f __server_port
f __INCOMING_MESSAGE_BUFFER_SIZE
f __ENCODING
```














server_bl.ServerBL

-  `__init__(self, host='localhost', port=50000, files_dir='.\downloads')`
-  `start(self)`
-  `__init_files_dir(self)`
-  `__handle_client_requests(self, client`
-  `__handle_request(self, client,`
-  `__handle_connect(self, client, params,`
-  `__handle_disconnect(self, client, params)`
-  `__handle_get_users(self, client, params)`
-  `__handle_send_msg(self, client, params)`
-  `__handle_send_msg_all(self, client, params)`
-  `__handle_get_list_file(self, client, params,`
-  `__handle_download(self, client, params)`
-  `__handle_proceed(self, client, params)`
-  `__handle_no_proceed(self, client, params)`
-  `__broadcast_message(self, message)`

-  `__host`
-  `__clientsManager_TCP`
-  `__clientsManager_UDP`
-  `__port`
-  `__server_TCP`
-  `__files_dir`
-  `__server_UDP`

server_bl.ClientsManager

-  `__init__(self)`
-  `add(self, nickname, client)`
-  `get_client_by_nickname(self, nickname`
-  `get_nickname_by_client(self, client`
-  `get_all_nicknames(self)`
-  `get_all_clients(self)`
-  `remove_by_nickname(self, nickname)`
-  `remove_by_client(self, client,`
-  `get_number_of_clients(self)`

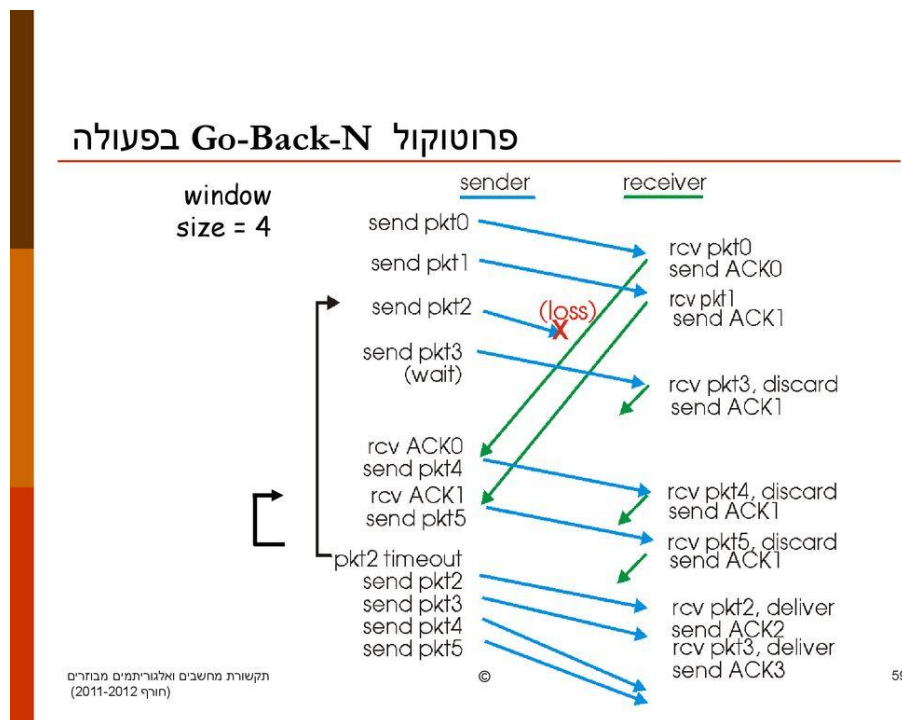
-  `__nicknames_to_clients`
-  `__clients_to_nicknames`

חלק ב'

כדי לשלוח את הקבצים נדרשנו לעבוד מעל פרוטוקול UDP ולוודא שלא היה איבוד חבילות.

דיאגרמת מצבים:

במימוש אצלינו גודל החלון הוא 10, לצורך הדוגמא נראה עם גודל חלון 4:



נסביר את הדיאגרמה. חבילות מס 0-3 נשלחות. חבילה מספר 2 נאבדה, חבילה מס 3 נקלטה אבל אנו מחכים ל ack מס 2 ולכן הוא שולח שוב את ack מס 1, ה timeout של חבילה מס 2 נגמר ולכן הוא שולח שוב את ה 4 חבילות מה ack האחרון שיש לו שהוא 1.

איך המערכת מתגברת על איבוד חבילות?

הגדרנו את גודל חלון שליחת החבילות להיות 10.

אנו שולחים 10 פאקטות ומחכים לקבל ack עליהם. אם חבילה נאבדה בדרך אז כשה timeout שלה ייגמר אנו שולחים שוב את ה 10 פאקטות מאותה חבילה.

איך נדע שהחבילה נאבדה? אנו מחכים לקבל לפי הסדר את מספר ה ack, כשאנו רואים שהמספר שהגיע אלינו זה לא המספר ack שאנו מחכים לקבל אנו זורקים את

החבילה ושולחים את ה ack הקודם, כך אנו מבינים שציפינו לקבל חבילה ולא קיבלנו אותה וכשייגמר ה timeout של החבילה שנאבדה אנו שולחים מאותה חבילה שוב.

נסביר את הדיאגרמה. חבילות מס 0-3 נשלחות. חבילה מספר 2 נאבדה, חבילה מס 3 נקלטה אבל אנו מחכים ל ack מס 2 ולכן הוא שולח שוב את ack מס 1, ה timeout של חבילה מס 2 נגמר ולכן הוא שולח שוב את ה 4 חבילות מה ack האחרון שיש לו שהוא 1.

איך המערכת מתגברת על בעיות latency?

המערכת מתגברת על בעיות latency ע"י זה שהקצבנו timeout לכל הודעה ובנוסף, הגבלנו את גודל החלון.