SHUTL

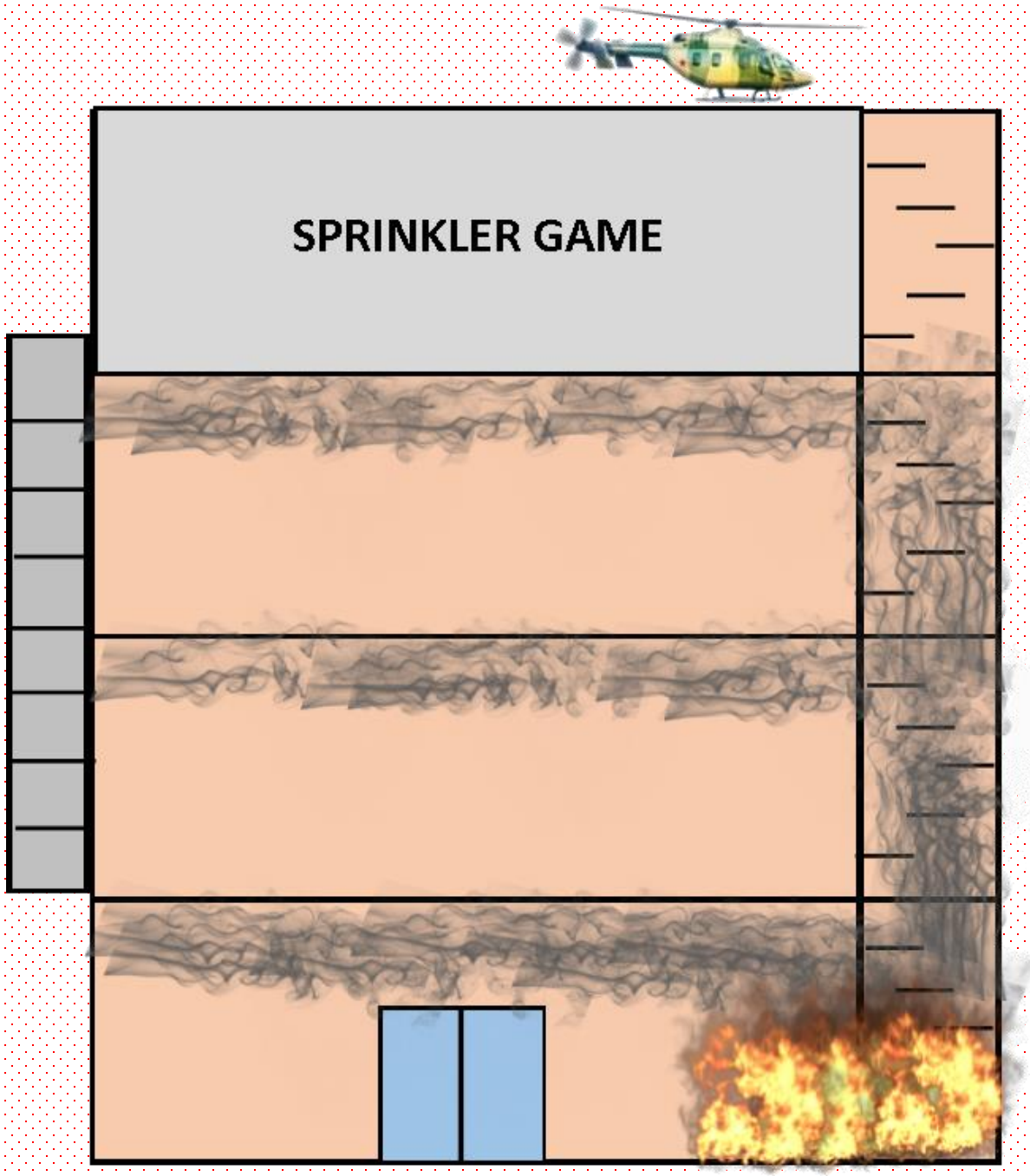
Created by Group 12

Brief Layout

- Instead of the typical flat layout of the map we flipped it on its head and turned it upside down. Our layout is 5 floors – Ground Floor, Floor 1, Floor 2, Floor 3 and the Roof.
- There are staircases to the right of the building and a fire escape to the left.
- There is a helicopter on the roof waiting to take you away with an injured pilot in there.
- There is a main door at the bottom that will give you access to the outside world and your freedom.
- There will be obstacles such fire and smoke that will hurt your health. For example if you enter the fire filled staircase, you will lose 50 hp
- Furthermore you will lose health every time you move up or down.

Items

- There are various different items in the game which will aid you within the game. These are…
  - Brick
  - Rope
  - Chocolate (Dairy Milk for Joe)
  - ID Card
  - Key
  - First Aid Kit
  - Fuel
  - Some of these will be very useful to your escape, but only if used in the correct place and the correct order



Visual Map

def eat_choc():

```
def eat_choc():
    """This function is responsible for adding to health
    when a chocolate bar is taken. If the health is 50 hp
    or above then health is reset. Otherwise 50 hp will be added.
    """

    global health

    if health >= 50:
        health = 100
    else:
        health += 50

    current_room["items"].remove(item_chocolate)
```

def health_is(health, room):

```
def health_is(health, room):
    """ This function is responsible for calculating the players health
    from their existing health and the damage attained from the room they
    are in. Accepts health as an integer value and room as a dictionary
    with a field damage."""

    damage = (room["damage"])
    return (health - damage)
```

def random_place_choc(rooms):

```
def random_place_choc(rooms):
    """This function is responsible for the random placement of the item item_chocolate.
    It loops through each of the rooms apart from the ground floor and the ground floor
    stairwell and selects a room at random in which to place the item.
    """

    potential_rooms = [
        "Roof",
        "Roof Stairwell",
        "Third",
        "Third Stairwell",
        "Third Fire Escape",
        "Second",
        "Second Stairwell",
        "Second Fire Escape",
        "First",
        "First Stairwell",
        "First Fire Escape"
    ]

    stop = len(potential_rooms)
    n = randrange(0, stop)
    room = rooms[potential_rooms[n]]
    room["items"].append(item_chocolate)
```

Inventory Functions

```
def find_in_inventory(inventory, item):
    '''Find a specific item by its id in the players inventory.
    Inventory must be passed as a list of dictionaries with each
    dictionary containing an id field. Item must be passed as a
    single dictionary containing an id field.
    '''

    for dictionary in inventory:
        if dictionary["id"] == item["id"]:
            return True
    return False


def clear_inventory():
    '''Removes all items in the players inventory.'''

    global inventory
    inventory = []
```

def find_required_items(inventory, required_items):

```
def find_required_items(inventory, required_items):
    '''This function takes a list of required item dictionaries and checks
    whether each item exists in the players inventory. If all the items are
    found the function will return True, otherwise it will return False.
    '''

    # For each item in required items check if it exists in inventory.
    # If it doesn't set match to False

    match = True

    for item in required_items:
        result = find_in_inventory(inventory, item)
        if not result:
            match = False

    # If there are no required items there is no match.

    if not required_items:
        match = False

    return match
```

Third Floor Game

```
def break_glass_game():

    seconds_to_run = 5
    required_hits = 40

    print("Do you wish to play?\n")

    while True:

        # Ask the user if they wish to play.
        print("To play, type YES. Otherwise, type NO.")
        play = input(">>> ")

        # If they want to play.
        if play.upper().strip() == "YES":

            # Count the number of hits.
            hits = count_hits(seconds_to_run)
            won = has_won(hits, required_hits)

            print('>>> %s hits' % hits)

            if won:
                remove_damage(rooms)
                print('YOU SMASHED THE GLASS')
                print(victory_string)
            else:
                print(failure_string)

            break

        # If they do not.
        elif play.upper().strip() == "NO":

            print("\nYou have chosen not to play.\n")
            break

        # If the input is not understood.
        else:

            print("\nI'm sorry, I didn't understand that.\n")
```

Why RetroBazinga should buy SHUTL

- Uniqueness
- Game in a game
- Challenging
- Need to use common sense
- Mystery
- Ready made Trailer

DEMO

Hope you enjoy the game!

- Hope you enjoy the game!
- By the way, there is a much easier way to escape from the building, you Should Have Used The Lift
- SHUTL