# Programming assignment

Group: AS1–2 Tony & DP1–1 Eric

Due: Feb.27.2020

# Table of contents

# Age determination

---

## Requirement

Five players are asked to give their name and their age.
The oldest and the youngest person has to be determined.
The output of the algorithm should say
"<name> is the oldest in the group, age <age>"
"<name> is the youngest in the group, age <age>”

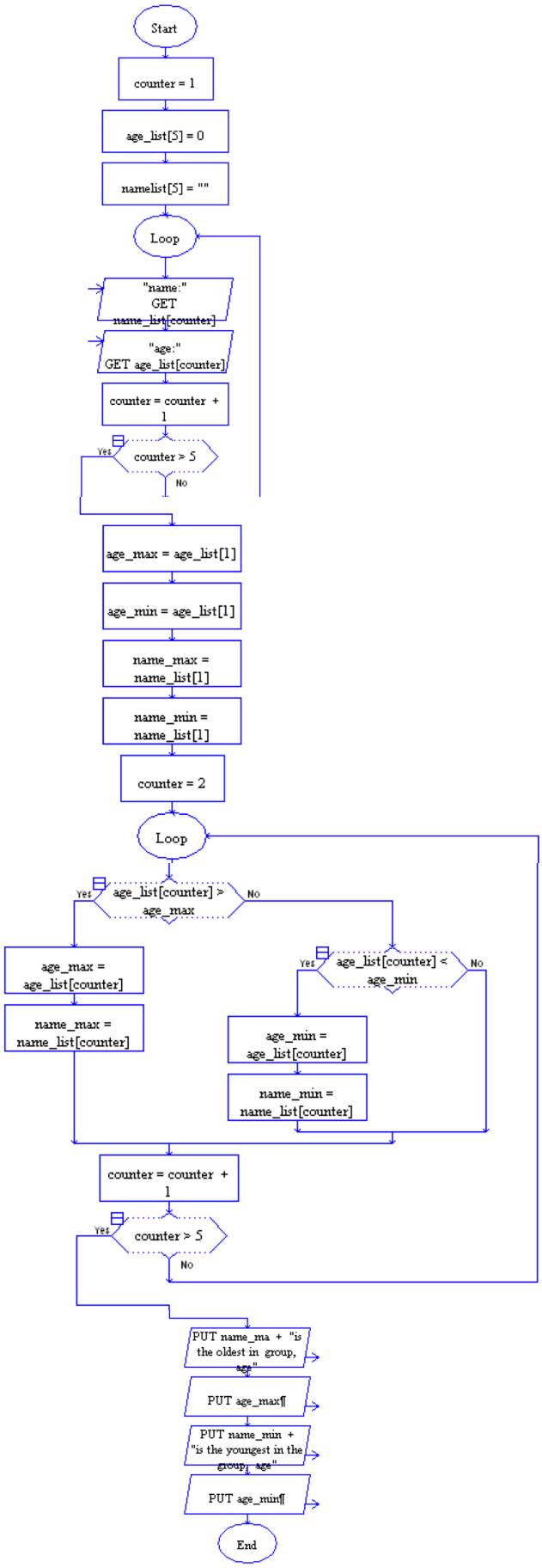
## Pseudocode

```
1.  DEFINE age_list[5]: INTEGER[]
2.  DEFINE name_list[5]: STRING[]
3.
4.  DEFINE counter: INTEGER
5.
6.  FOR counter <- 0 TO 5
7.      name_list[counter] <- INPUT("Player", counter + 1, ", please input your
    name.")
8.      age_list[counter] <- INPUT("Player", counter + 1, ", please input your age.")
9.  ENDFOR
10.
11. DEFINE age_max: INTEGER
12. DEFINE age_min: INTEGER
13. DEFINE name_max: STRING
14. DEFINE name_min: STRING
15.
16. age_max = age_list[0]
17. age_min = age_list[0]
18.
19. name_max = name_list[0]
20. name_min = name_list[0]
21.
22. FOR counter <- 1 TO 5
23.     IF age_list[counter] > age_max THEN
24.         age_max = age_list[counter]
25.         name_max = name_list[counter]
26.     ELSE IF age_list[counter] < age_min THEN
27.         age_min = age_list[counter]
28.         name_min = name_list[counter]
29.     ENDIF
30. ENDFOR
31.
32. OUTPUT (name_max, " is the oldest in the group, age ", age_max)
33. OUTPUT (name_min, " is the youngest in the group, age ", age_min)
```

## Identifier table

| Variable | Data type | Description |
|---|---|---|
| age_list | integer[] | store five players’ ages |
| name_list | string[] | store five players’ names |
| counter | integer | counter |
| age_max | integer | store the oldest person’s age |
| age_min | integer | store the youngest person’s age |
| name_max | string | store the oldest person’s name |
| name_min | string | store the youngest person’s name |

# Raptor

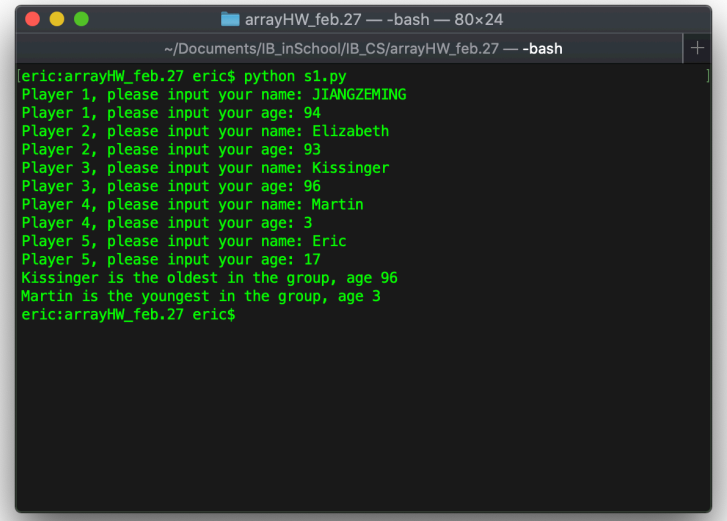

Start
counter = 1
age_list[5] = 0
namelist[5] = ""
Loop
"name:"
GET
name_list[counter]
"age:"
GET age_list[counter]
counter = counter + 1
Yes
counter > 5
No
age_max = age_list[1]
age_min = age_list[1]
name_max = name_list[1]
name_min = name_list[1]
counter = 2
Loop
Yes
age_list[counter] > age_max
No
age_max = age_list[counter]
name_max = name_list[counter]
Yes
age_list[counter] < age_min
No
age_min = age_list[counter]
name_min = name_list[counter]
counter = counter + 1
Yes
counter > 5
No
PUT name_ma + "is the oldest in group, age"
PUT age_max¶
PUT name_min + "is the youngest in the group, age"
PUT age_min¶
End

## Python

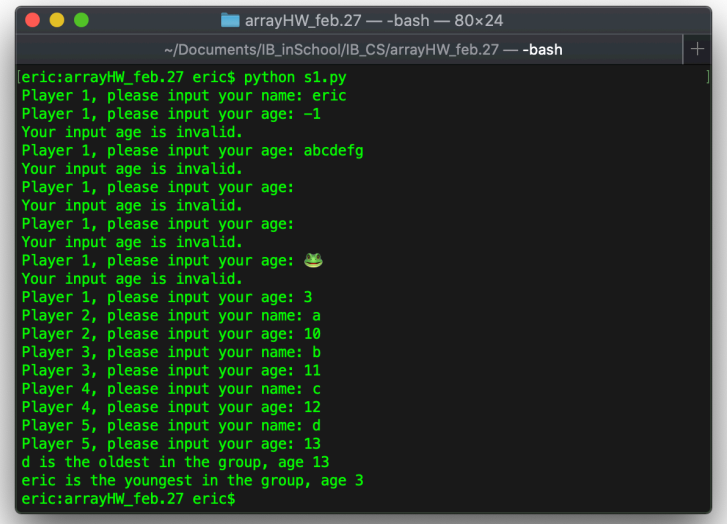
```python
# Assume that no two players have same age.

age_list = []
name_list = []

for i in range(5):
    name = input("Player " + str(i + 1) + ", please input your name: ")
    while True:
        try:
            age = int(input("Player " + str(i + 1) + ", please input your age: "))
            if age < 0:
                raise(Exception)
            else:
                break
        except:
            print("Your input age is invalid.")

    age_list.append(age)
    name_list.append(name)

age_max = age_list[0]
age_min = age_list[0]

name_max = name_list[0]
name_min = name_list[0]

for i in range(1, 5):
    if age_list[i] > age_max:
        age_max = age_list[i]
        name_max = name_list[i]
    elif age_list[i] < age_min:
        age_min = age_list[i]
        name_min = name_list[i]

print(name_max, "is the oldest in the group, age", age_max)
print(name_min, "is the youngest in the group, age", age_min)
```

# Testing

## Normal testing:




## Idiot proof:



# Game

---

## Requirement

3 players role two dice at the same time.
The winner has to be displayed and determined based on the following decision tree:
If the first die of any player is the highest, he wins. The second die doesn't count.
If there is a tie in the score of the first die, add the second die to the total score per player. Highest overall wins.
If there is still not winner according to these condition, then the winner will be determined based on score die 1 = score die 2, you win and if this is not the case you will show "no winner".

## Pseudocode

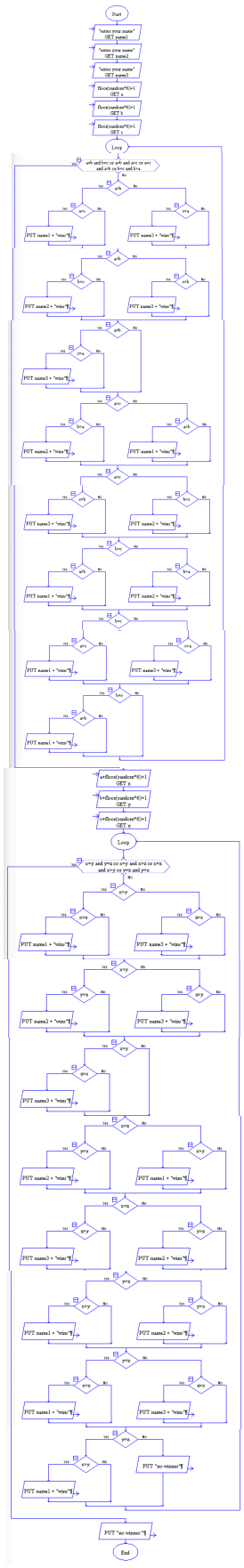
```
1.  DEFINE die_1_nums: INTEGER[3]
2.  DEFINE die_2_nums: INTEGER[3]
3.  DEFINE name_list: STRING[3]
4.  die_1_nums <- []
5.  die_2_nums <- []
6.  name_list <- []
7.  DEFINE counter: INTEGER
8.  counter <- 0
9.
10. FOR counter <- 0 to 3
11.     die_1_nums[counter] <- randint(1,6)
12.     die_2_nums[counter] <- randint(1,6)
13.     name_list[counter] <- INPUT("Give your name")
14. ENDFOR
15.
16. DEFINE num_max: INTEGER
17. DEFINE name_max: STRING
18. DEFINE tie: BOOL
19.
20. num_max <- die_1_nums[0]
21. name_max <- name_list[0]
22. tie <- FALSE
23.
24. FOR counter <- 1 to 3
25.     IF die_1_nums[counter] > num_max THEN
26.         num_max = die_1_nums[counter]
27.         name_max = name_list[counter]
28.         tie = FALSE
29.     ELSE IF die_1_nums[counter] == num_max THEN
30.         tie = TRUE
31.     ENDIF
32. ENDFOR
33.
34. IF tie THEN
35.     DEFINE die_sum_nums: INTEGER[3]
36.     die_sum_nums <- []
37.     FOR counter <- 0 to 3
38.         die_sum_nus[counter] <- die_1_nums[counter] + die_2_nums[counter]
39.     ENDFOR
40.
41.     num_max <- die_sum_nums[0]
42.     name_max <- name_list[0]
43.     tie = FALSE
44.
45.     FOR counter <- 1 to 3
46.         IF die_sum_nums[counter] > num_max THEN
47.             num_max = die_sum_nums[counter]
48.             name_max = name_list[counter]
49.             tie = FALSE
50.         ELSE IF die_sum_nums[counter] == num_max THEN
51.             tie = TRUE
52.         ENDIF
53.     ENDFOR
54.
55.     IF tie THEN
56.         FOR counter <- 0 to 3
57.            IF die_1_nums[counter] == die_2_nums[counter] THEN
58.                 OUTPUT name_list[counter], " win."
59.                 EXIT
60.             ENDIF
61.         ENDFOR
62.         OUTPUT "no winner"
63.     ELSE
64.         OUTPUT name_max + " win."
65.     ENDIF
66.
67. ELSE
68.     OUTPUT name_max + " win."
69. ENDIF
```

## Identifier table

| Variable | Data type | Description |
|---|---|---|
| name_list | string[] | players’ names |
| die_1_nums | int[] | players’ 1st die numbers |
| die_2_nums | int[] | players’ 2nd die numbers |
| tie | bool | tie or not |
| name_max | string | player with the highest number’s name |
| num_max | int | highest number |
| die_sum_nums | int[] | sums of 1st and 2nd number |
| counter | int | counter |

# Raptor

*NOTE: Algorithm presented by Raptor is inconsistent with the one presented in pseudocode & python*



## Python

```python
from random import randint

if __name__ == "__main__":
    die_1_nums = []
    die_2_nums = []
    name_list = []

    for i in range(3):
        die_1_nums.append(randint(1, 6))
        die_2_nums.append(randint(1, 6))
        name_list.append(iput("player " + str(i + 1) + ", please input your name: "))

    num_max = die_1_nums[0]
    name_max = name_list[0]
    tie = False

    for i in range(1, 3):
        if die_1_nums[i] > num_max:
            num_max = die_1_nums[i]
            name_max = name_list[i]
            tie = False
        elif die_1_nums[i] == num_max:
            tie = True

    if tie:
        diesum_nums = [die_1_nums[i] + die_2_nums[i] for i in range(3)]
        num_max = die_sum_nums[0]
        name_max = name_list[0]
        tie = False

        for i in range(1, 3):
            if die_sum_nums[i] > num_max:
                num_max = die_sum_nums[i]
                name_max = name_list[i]
                tie = False
            elif die_sum_nums[i] == num_max:
                tie = True

        if tie:
            for i in range(3):
                if (die_1_nums[i] == die_2_nums[i]):
                    print(name_list[i], "win.")
                    break
            else:
                print("no winner.")
        else:
            print(name_max, "win.")
    else:
        print(name_max, "win.")
```

## Testing

**Win by 1st die number:**

IO:

```
player 1, please input your name: a
player 2, please input your name: b
player 3, please input your name: c
a win.
```

Random variables:

```
> die_1_nums: [6, 5, 5]
> die_2_nums: [6, 1, 5]
  i: 2
> name_list: ['a', 'b', 'c']
```

**Win by sum die number:**

IO:

```
player 1, please input your name: a
player 2, please input your name: b
player 3, please input your name: c
c win.
```

Random variables:

```
> die_1_nums: [2, 1, 2]
> die_2_nums: [3, 4, 5]
  i: 2
> name_list: ['a', 'b', 'c']
```

**Win by same die number:**

IO:

```
player 1, please input your name: a
player 2, please input your name: b
player 3, please input your name: c
b win.
```

Random variables:

```
> die_1_nums: [3, 3, 1]
> die_2_nums: [2, 3, 5]
  i: 2
> name_list: ['a', 'b', 'c']
```

**No winner:**

IO:

```
player 1, please input your name: a
player 2, please input your name: b
player 3, please input your name: c
no winner.
```

Random variables:

```
> die_1_nums: [3, 2, 3]
> die_2_nums: [5, 1, 5]
  i: 2
> name_list: ['a', 'b', 'c']
```