10 MASTERMIND

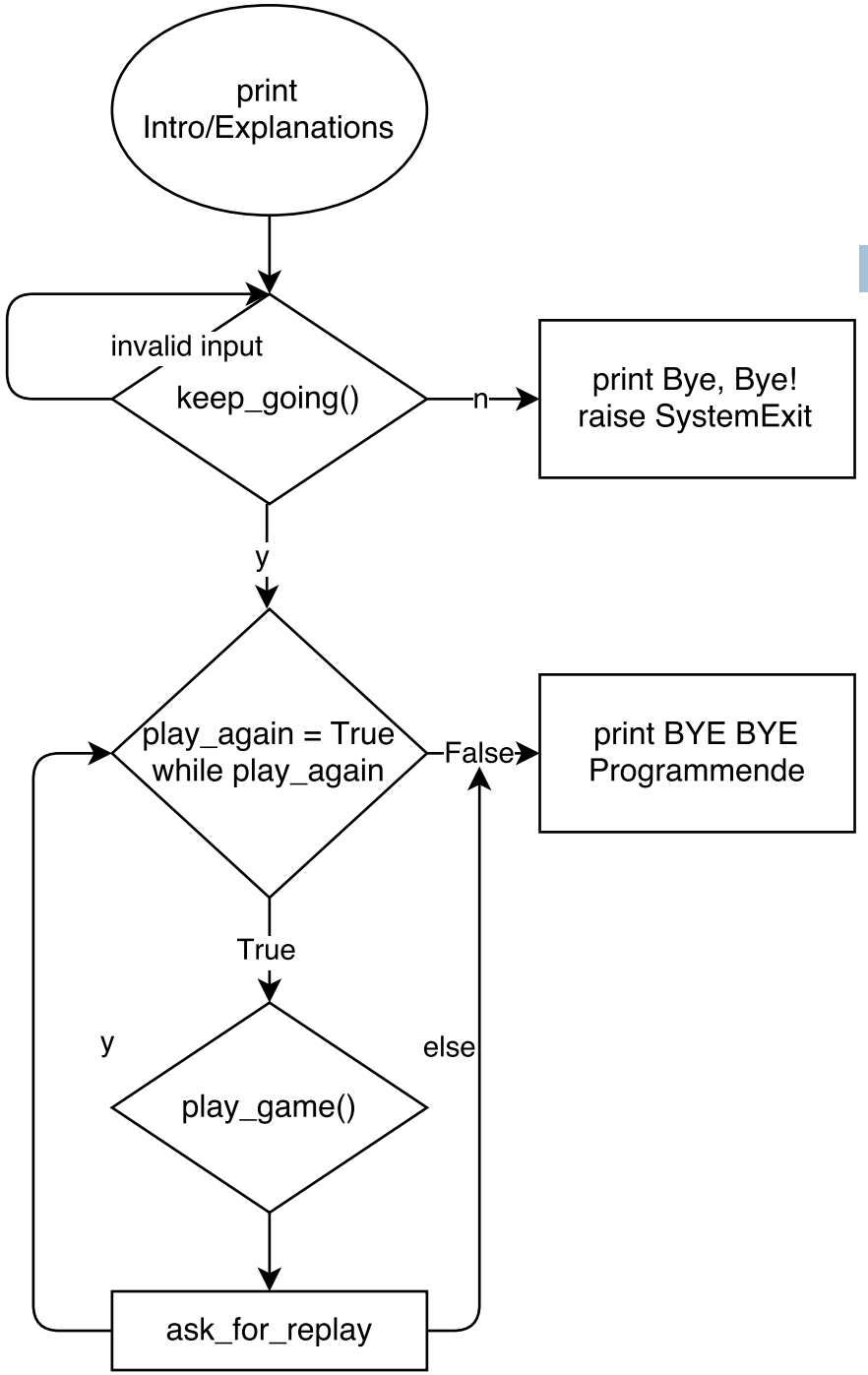
Yanick DICKBAUER, Patrick MOSER, Manuel PERNER

Aufgabenstellung

- Simulieren Sie das Spiel “Mastermind”. Der Computer soll zufällig einen vierstelligen Zahlencode generieren, wobei die Zahlen sechs unterschiedliche Werte annehmen können. Der Spieler muss nun den vierstelligen Zahlencode erraten. Der Computer teilt ihm mit, wie viele richtige Zahlen er an der richtigen Position erraten hat und wie viele richtige Zahlen er an der falschen Position erraten hat. Danach kann der Spieler nochmals raten, bis er die richtige Kombination gefunden hat. Je nach Anzahl der benötigten Schritte gibt der Computer einen Kommentar ab, nach einer gewissen Anzahl an Versuchen bricht er ab und gibt die richtige Kombination aus.
- Stellen Sie im Rahmen der Präsentation den Ablauf des Programmes anhand von selbstgewählten Zufallszahlen vor und zeigen Sie die Auswertungmethode einer Runde anhand von unterschiedlichen Beispielen.

Flow Chart

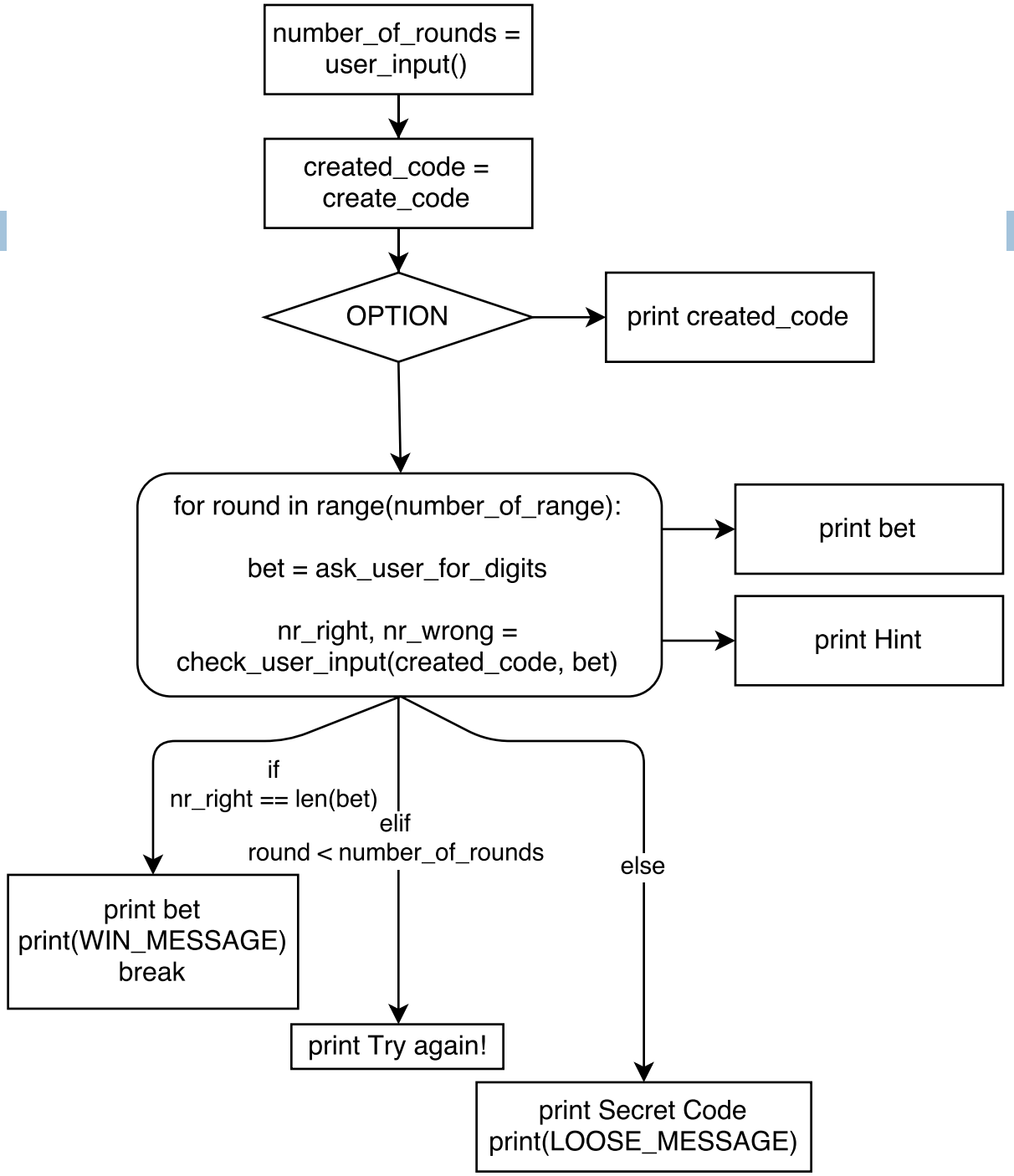
MAIN FUNCTIONALITY



Flow Chart

FUNCTION

play_game()



FUNCTION play_game()

```
81  def play_game():
82      number_of_rounds = user_input([
83          ['Please enter the number of rounds', int, 10]], DEBUG)[0]
84
85      created_code = create_code()   #given by a defined variable
86      if OPTION:
87          # show the right answer:
88          print('Right answer would be:', created_code)
89
90      for round in range(number_of_rounds):
91          bet = ask_user_for_digits()
92          #bet = [4, 2, 0, 0]
93          print('Your bet:',bet)
94          nr_right, nr_wrong = check_user_input(created_code, bet)
95          print('Hint:', 'o'*nr_right,'x'*nr_wrong)
96          if nr_right == len(bet):
97              print('Your guess of ',bet,' equals the secret code!')
98              print(WIN_MESSAGE)
99              break
100         elif round < number_of_rounds-1:
101             print('Try again!')
102         else:
103             print('The secret code was:', created_code)
104             print(LOOSE_MESSAGE)
```

FUNCTION create.code()

- erzeugt einen Zahlencode
  - benutzt für created_code
  - inkl. DEBUG-Option für DEMO

```
106 ▼ def create_code():
107       """This function creates a random 4-digit code out of the numbers 1 to 6"""
108       code = []
109       for i in range(4):
110           code.append(int(random_number_from_interval(1,6+1)))
111       if DEBUG:
112           return [4,5,4,2]
113       return code
```

FUNCTION ask_user_for digits()

- holt sich den User-Tipp
  - inkl. Eingabeüberprüfung (valid_input)

```
143 def ask_user_for_digits():
144     valid_input = False
145     user_input = input('Please enter 4 digits: ')
146     while valid_input == False:
147         digits = []
148         for i, char in enumerate(user_input):
149             if char in '123456':
150                 digits.append(int(char))
151             else:
152                 break
153         if len(digits) == 4:
154             valid_input = True
155         else:
156             user_input = input('Make sure to enter 4 integers between 1 and 6: ')
157     return digits
```

FUNCTION

check_user_input(right_digits, user_digits)

```
116 def check_user_input(right_digits, user_digits):
117     """Returns two values: (nr_correct_on_pos, nr_correct_on_wrong_pos)
118     i.e. nr_correct_on_pos is the number of correct values on the right pos
119          nr_correct_on_wrong_pos is the --.-- on the wrong position"""
120     nr_right_pos = 0
121     nr_wrong_pos = 0
122     right_digits_modified = right_digits[:]
123 
124     # check for right digits on right positions:
125     for i, act_digit in enumerate(user_digits):
126         if right_digits[i] == act_digit:
127             nr_right_pos += 1
128             # block found digits to make sure we dont count them twice on wrong pos
129             right_digits_modified[i] = 'x'
130 
131     # check for digits on wrong pos:
132     for i, act_digit in enumerate(user_digits):
133         # skip those which we already marked as right pos:
134         if right_digits[i] != act_digit:
135 
136             if act_digit in right_digits_modified:
137                 nr_wrong_pos += 1
138                 index_of_right_mod = right_digits_modified.index(act_digit)
139                 right_digits_modified[index_of_right_mod] = 'x'
140 
141     return nr_right_pos, nr_wrong_pos
142 
```

FUNCTION main()

```
67 ▼ def main():
68       print(INTRO)
69       keep_going()
70
71       play_again = True
72 ▼     while play_again:
73           play_game()
74           ask_for_replay = input()
75           if ask_for_replay == 'y':
76               play_again
77 ▼         else:
78               play_again = False
79               print('                                    BYE BYE')
80
```