Level: 1st year of computer science
Course: ADS1

Typical solution

Academic year:2024/2025
Duration: 1h :30m

Exercise 1 : (5 pts)

Consider the following program:

- Correct the four syntactic errors present in the code. (1 pts)
- Run the program. (give values of each variable at the end) (1 pts)

i=3, j=8, p=1, c=3

- What does this code do? (1 pts)

The program identifies and prints the index of the first row in the matrix that contains an odd number of 1s.

- Rewrite lines 8 to 12 in algorithmic form. (2 pts)

```
p←0
i←0
while ( i < 3 && p==0)
 c←0
 for j←0 to 7 do
  if (A[i][j] = 1)
   c←c+1
  end if
 end for
 p←c mod 2
 i←i+1
end while
```

```
1  #include <stdio.h>
2  int main() {
3   int i, j, p, c;
4   int A[3][8] = {
5       {1, 1, 0, 1, 1, 0, 1, 1},
6       {1, 0, 0, 1, 0, 1, 1, 0},
7       {0, 1, 0, 0, 1, 1, 0, 0}};
8   for (p=0,i = 0; i < 3 && p==0; i++){
9    for (c=0,j = 0; j < 8; j++)
10    if (A[i][j] == 1) ++c;
11   p=c % 2;
12  }
13  if (p!=0)
14   printf("%d\n", i-1);
    return 0;
}
```

Exercise 2 : (4.5 pts)

```
#include <stdio.h>
int main() {
 int i, j, n;
 printf("Enter the number of rows for the
triangle: ");
 scanf("%d", &n);
 for (i = 1; i <= n; i++) {
  for (j = 1; j <= n - i; j++)
    printf(" ");
  for (j = 1; j <= 2 * i - 1; j++)
    printf("*");
  printf("\n");
 }
 return 0;
}
```

```
Algorithm triangle للفائدة فقط
 var i, j, n: integer
begin
 write("Enter the number of rows for the
triangle:")
 read(n)
 for i←1 to n do
  for j←1 to n-i do
   write(" ")
  end for
  for j←1 to 2*i-1 do
   write("*")
  end for
  write("\n")
end for
end
```

Exercise 3 : (5.5 pts)

```
Algorithm Separate_Digits
 var n, digit, i, odd_count, even_count:
integer
 odd_digits, even_digits: array[10] of
integer
begin
 do
  write("enter an integer n > 1000:")
  read(n)
 while n<1000
 odd_count ← 0
```

```
#include <stdio.h> للفائدة فقط
int main() {
  int i, n, digit, odd_count=0,
even_count=0,  odd_digits[10],
even_digits[10];
  do{
   printf("Enter an integer n > 1000: ");
   scanf("%d", &n);
  }while n<1000;
  while (n > 0) {
    digit = n % 10;
```

```
 even_count ← 0
 while n > 0 do
  digit ← n mod 10
  if digit mod 2 = 0 then
   even_digits[even_count] ← digit
   even_count ← even_count + 1
  else
   odd_digits[odd_count] ← digit
   odd_count ← odd_count + 1
  end if
  n ← n div 10
 end while

 write("odd digits:")
 for i ← 0 to odd_count - 1 do
  write(odd_digits[i])
 end for

 write("even digits:")
 for i ← 0 to even_count - 1 do
  write(even_digits[i])
 end for
end
```

```
    if (digit % 2 == 0) {
      even_digits[even_count] = digit;
      even_count++;
    } else {
      odd_digits[odd_count] = digit;
      odd_count++;
    }
    n /= 10;
  }

  printf("Odd digits: ");
  for (i = 0; i < odd_count; i++)
    printf("%d ", odd_digits[i]);
  printf("\nEven digits: ");
  for (i = 0; i < even_count; i++)
    printf("%d ", even_digits[i]);
  return 0;
}
```

Exercise 4 : (5 pts)

```
#include <stdio.h>
Typedef struct {
  char name[50];
  int time, difficulty;
} Recipe;

int main() {
  int n, i, specific_difficulty, max_time;
  Recipe recipes[50];
  printf("Enter the number of recipes: ");
  scanf("%d", &n);
  for (i = 0; i < n; i++) {
    printf("Enter recipe name: ");
    gets(recipes[i].name);
    printf("Enter cooking time (in minutes): ");
    scanf("%d", &recipes[i].time);
    printf("Enter difficulty level (1-5): ");
    scanf("%d", &recipes[i].difficulty);
  }

  printf("Enter the specific difficulty level (1-5): ");
  scanf("%d", &specific_difficulty);
  printf("Enter the maximum cooking time (in minutes): ");
  scanf("%d", &max_time);

  printf("\nRecipes that match the criteria:\n");
  for (i = 0; i < n; i++)
    if (recipes[i].difficulty == specific_difficulty && recipes[i].time <= max_time)
      printf("Recipe: %s, Cooking Time: %d minutes\n",
          recipes[i].name, recipes[i].time, recipes[i].difficulty);
  return 0;
}
```