C3 documentation

User documentation

Task

We have the daily midday temperature measurements from the previous N days.
Write a program that gives the longest period of days when the temperature was always above K(unknown input) degrees.

Runtime environment

An lenovo pc that is capable of running exe files,64 -bit operating system .No mouse needed

Usage

Program input

The program reads the input data from the keyboard in the following order:

#	data	**explainantion**
1.	N	the count of days $(1 \leq N \leq 100)$
2.	K	temperature $(-20 \leq K \leq 50)$
3.	$temp_1$	First day temp $(20 \leq Temp_1 \leq 50)$
4.	$temp_2$	Second day temp $(20 \leq Temp_2 \leq 50)$
..	..	
5.	$temp_i$	The Nth temp $(20 \leq Temp_N \leq 50)$

Program output

The program writes out the starting and ending identifiers of the longest period of days , which the temperature was greater than K(unknown input) .

Sample input and output

#input
#count of days
1. N[1..100] = 7
temperature
1. K[-20..50] = 30
#days temperatures
1. $temp_1$[20..50] = 25
2. $temp_2$[20..50] = 36
3. $temp_3$[20..50] = 29
4. $temp_4$[20..50] = 33
5. $temp_5$[20..50] = 34
6. $temp_6$[20..50] = 36
7. $temp_7$[20..50] = 30
Output
Start = 4
Last1 = 6

Possible errors

The input should be given according to the sample. If the count of days or temperature is not a whole number, or the range of count of days is not from 1..100 or the range of temperature is not from -20..50 , it will cause a problem. If one of the days temperatures is not a whole number , or not a number, or it is not in the range 20..50, it also will cause a problem. In the case of an error, the program displays an error message and asks for the repetition of the input.

Sample of running in the case of invalid data:

N[1..100] = 0
N[1..100] = 1.5
N[1..100] = something
N[1..100] = 50
K[-20..50] = -100
K[-20..50] = 1.5
K[-20..50] = something
K[-20..50] = 30
temp$_1$[20..50] = 10
temp$_1$[20..50] = abs
temp$_1$[20..50] = 5.5
temp$_1$[20..50] = -10
temp$_1$[20..50] = 30
temp$_2$[20..50] = 40
temp$_3$[20..50] = 45

Developer documentation

Task

We have the daily midday temperature measurements from the previous N days.
Write a program that gives the longest period of days when the temperature was always above K(unknown input) degrees.

Specification

Input: $N \in \mathbb{N}$; $K \in Z$; $temp_{0..N-1} \in N^N$

Output: Start $\in N$; last1 $\in N$

Precondition: $\forall i\,(0 \leq I \leq N-1)$: $temp_i \in [20..50]$ and $N \in [1..100]$ and $k \in [-20..50]$

Postcondition:

$$Cnt = \sum_{\substack{I=0 \\ Temp[i]>k}}^{n-1} 1$$

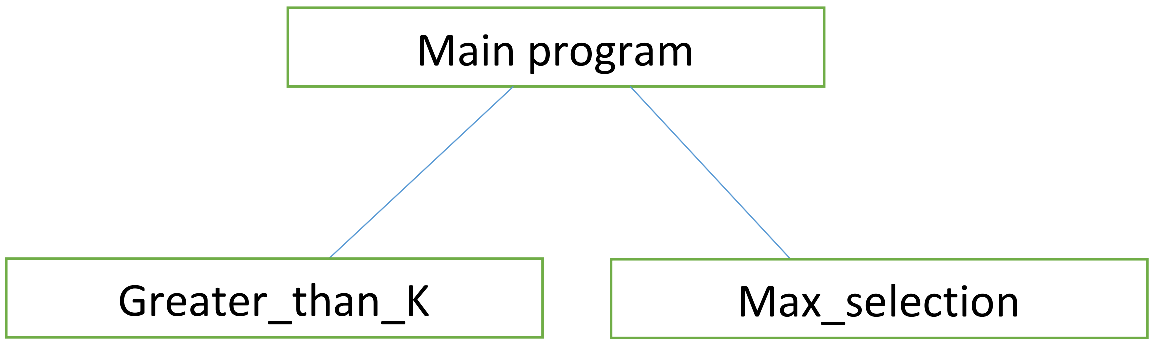
$Result_{0..cnt-1} \in N^{cnt}$ $\forall i(0 \leq I \leq cnt-1)$: $result_i > k$
Len $\in$ N ; maxlen $\in$ N
If $\exists i(1 \leq I \leq cnt-1)$: $result_i = result_{i-1} + 1$ -> len = len + 1
Otherwise len = 1
And if len > maxvalue -> last1 = $result_i$ and maxlen = len
Start = (last1 - maxlen) + 1

Comment: If there is no days greater than K temperature the program will cout “nincs”

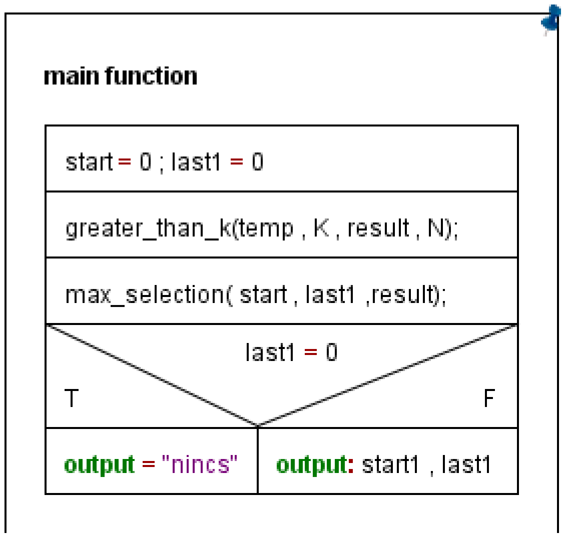
Developer environment

lenovo PC, an operating system capable of running exe files .mingw32-g++.exe c++
compiler (v4.7), Code::Blocks (v13.12) developer tool.

Functions structure

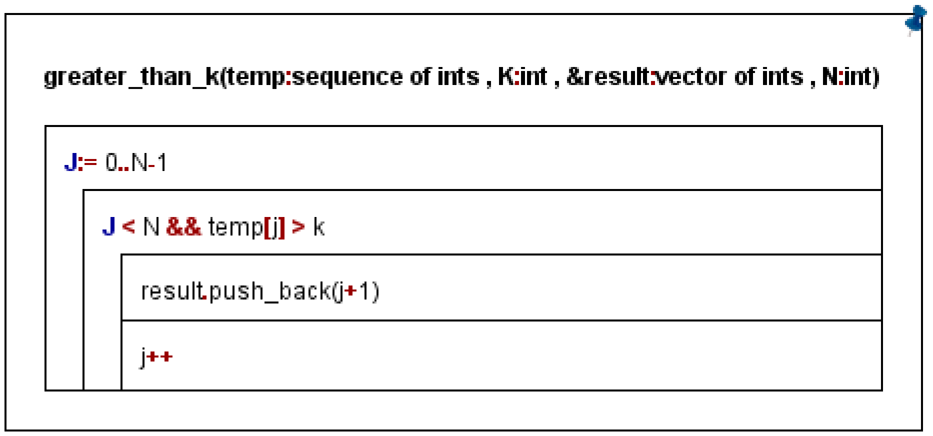


Main program

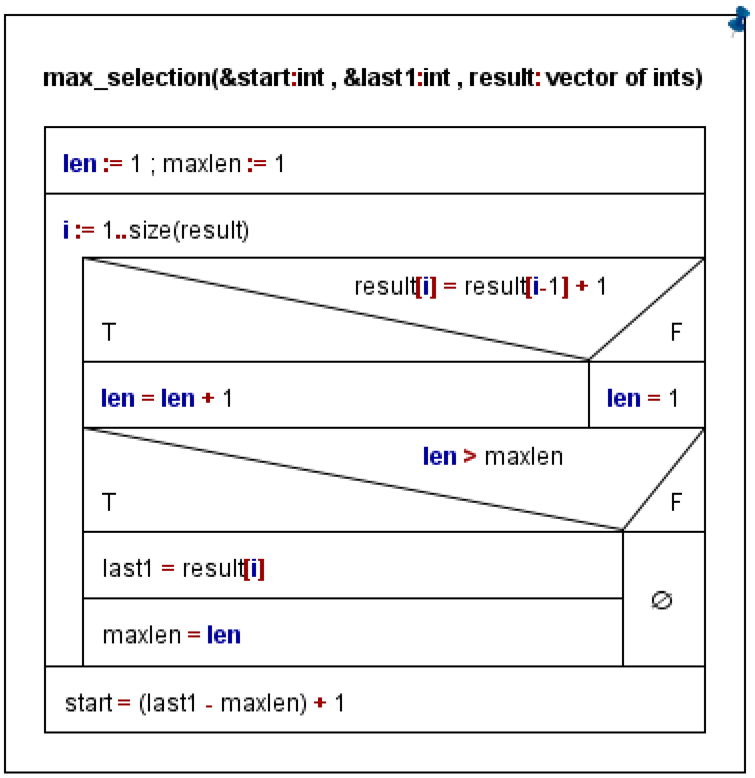


```
start = 0 ; last1 = 0
greater_than_k(temp , K , result , N);
max_selection( start , last1 ,result);
last1 = 0
output = "nincs"
output: start1 , last1
```

Subprograms



```
greater_than_k(temp:sequence of ints , K:int , &result:vector of ints , N:int)
J:= 0..N-1
    J < N && temp[j] > k
        result.push_back(j+1)
        j++
```



```
max_selection(&start:int , &last1:int , result: vector of ints)
len := 1 ; maxlen := 1
i := 1..size(result)
    result[i] = result[i-1] + 1
    T:
        len = len + 1
    F:
        len = 1
    len > maxlen
    T:
        last1 = result[i]
        maxlen = len
    F:
        ∅
start = (last1 - maxlen) + 1
```

Code

```
#include <iostream>

#include <vector>

using namespace std;

// student : Omar Ashour

// neptun code : e5dzst

void greater_than_k(int temp[] , int K , vector<int>& result , int N);

void max_selection(int& start , int& last1 , vector<int> result );


int main()

{int N , K;

vector<int> result;

vector <int> last;

int start = 0;

int last1 = 0;

bool repeat = true;

 while (repeat) {

      cin >> N;

      repeat = cin.fail() || cin.peek()!='\n' || N < 1 || N > 100;

      if (repeat) cout << "You entered an invalid number." << endl;

      cin.clear();

      cin.ignore(999, '\n');


}

repeat = true;

 while (repeat) {

      cin >> K;

      repeat = cin.fail() || cin.peek()!='\n' || K < -20 || K > 50;

      if (repeat) cout << "You entered an invalid number." << endl;

      cin.clear();

      cin.ignore(999, '\n');


}

int temp [N];
```

```
for(int i = 0 ; i < N ; i++){
   repeat = true;
 while (repeat) {
      cin >> temp[i];
      repeat = cin.fail() || cin.peek()!='\n' || temp[i] < 20 || temp[i] > 50;
      if (repeat) cout << "You entered an invalid number." << endl;
      cin.clear();
      cin.ignore(999, '\n');

}
}
greater_than_k(temp , K , result , N);
max_selection( start , last1 ,result);
if(last1 == 0){
   cout << "nincs";
}
else{
    cout << start << " "<< last1 << endl;
}}

void greater_than_k(int temp[] , int K , vector<int>& result , int N){
for(int j = 0 ; j < N ; j++){
while(j < N && temp[j] > K){
   result.push_back(j+1);
   j++;
}
}


}
void max_selection(int& start , int& last1 , vector<int> result ){
int len = 1;
int maxlen = 1;
for( unsigned int i = 1 ; i < result.size() ; i++){


   if(result[i] == result[i-1] + 1) len++;
   else  len = 1;
   if(len > maxlen){
 last1 = result[i];
      maxlen = len;

   }
}
start = (last1 - maxlen) + 1;

}
```

Testing

Valid test cases

1 . test case : input 1.txt

Input- no sequence
N = 1
K = 30
$Temp_1$ = 31
Output
“nincs”

2. test case : input 2.txt

Input – big sequence with solution
N = 7
K = 30
$Temp_0$ = 25
$Temp_1$ = 36
$Temp_2$ = 29
$Temp_3$ = 33
$Temp_4$ = 34
$Temp_5$ = 36
$Temp_6$ = 30
Output
Start = 4
Last1 = 6

3. test case : input 3.txt

Input- small sequence with solution
N = 2
K = 30
$Temp_1$ = 31
$Temp_2$ = 32
Output
Start = 1
Last1 = 2

4. test case : input 4.txt

Input- small sequence with no solution
N = 2
K = 30
$Temp_1$ = 20
$Temp_2$ = 21
Output
“nincs”

5. test case : input 5.txt

Input- big sequence with no solution
N = 5
K = 30
$Temp_1$ = 25
$Temp_2$ = 26
$Temp_3$ = 27
$Temp_4$ = 28
$Temp_5$ = 29
Output
“nincs”

6. test case : input 6.txt

Input- incorrect input: count of days(N)
N = a7a or N = 1.1 or N = 101
“You entered an invalid number.”
Ask again :
N = …

7. test case : input 7.txt

Input- incorrect input: temprature(K)
N = 50
K= a8a or K = 5.5 or K = 501
“You entered an invalid number.”
Ask again :
K = …

8. test case : input 8.txt

Input- incorrect input: days tempratures($Temp_i$)
N = 50
K = 30
$Temp_1$= a8a or $Temp_1$ = 1.5 or $Temp_1$ = 19
“You entered an invalid number.”
Ask again :
$Temp_1$= …

9. test case : input9.txt

Input- incorrect input: days tempratures($Temp_i$)
N = 50
K = 30
$Temp_1$= a8a
“You entered an invalid number.”
Ask again :
$Temp_1$= …