be-OI 2011 **Finale** 30 maart 2011	**Gegevens invullen in HOOFDLETTERS en LEESBAAR, aub** VOORNAAM+NAAM : . . . . . . SCHOOL : . . . . . . ZAAL : CANDIX / DAO COMPUTER N°	**Gereserveerd**

Belgische Olympiades in de Informatica (duur : 2u maximum)

Dit is de vragenlijst van het **gedeelte aan de computer** van de finale van de Belgische Olympiades in de Informatica voor de **categorie hoger onderwijs**. Er zijn 3 vragen. De eerste vraag staat niet op punten maar dient om je vertrouwd te maken met het systeem gedurende het eerste **kwartier**. De twee volgende vragen worden pas uitgedeeld nadat dit kwartier is afgelopen, en voor die vragen krijg je dan **maximum 1u45 de tijd**.

Enkel de code die je via de website indient zal meetellen bij de beoordeling van dit onderdeel.

Af te geven

1. Je programma leest parameters en gegevens via de standaard input en moet het resultaat naar de standaard output schrijven. Je moet je programma indienen via het indieningssysteem. Alle details daarover staan op de volgende pagina.
2. Je moet ook je vragenlijst teruggeven, met de kader met gegevens bovenaan op het eerste blad correct ingevuld.

Algemene opmerkingen (lees dit aandachtig voor je begint)

1. Schrijf je naam, voornaam, school, **naam van de zaal en nummer van de computer** op de eerste pagina. Leg je studentenkaart of identiteitskaart op de tafel.
2. Ga zitten op de **plaats** die je wordt **toegewezen** door de organisatoren.
3. Je mag alleen iets om de schrijven bij je hebben. Rekenmachines, GSM, ... zijn **verboden**. Laat je persoonlijke bagage achter op de plaats die wordt aangeduid door de toezichthouders.
4. Je mag **op geen enkel moment met iemand communiceren**, behalve met de organisatoren of toezichthouders. Alle technische en inhoudelijke vragen mag je enkel stellen aan de organisatoren. Logistieke vragen mogen gesteld worden aan de toezichthouders.
5. Je hebt **geen internettoegang** tijdens de proef. Elke poging om andere deelnemers of buitenstaanders te contacteren zal worden bestraft.
6. Je **mag** alle functionaliteit gebruiken van de standaardbibliotheken van de taal die je kiest (Java, C, C++, Pascal, Python of PHP), behalve alles dat communicatie met de buitenwereld impliceert. Enkel de standaard input en output zijn daarvoor toegelaten. In de praktijk betekent dat dat je geen toegang tot een netwerk mag maken, en ook niet mag lezen uit of schrijven naar bestanden op de harde schijf.
7. Je mag **kladbladen** vragen aan de toezichthouders.
8. Het is strikt **verboden te eten of te drinken** in de computerlokalen. Je mag je **plaats niet verlaten tijdens de proef**, bijvoorbeeld om naar het toilet te gaan of te roken.
9. Je krijgt **exact 2 uur** om deze proef op te lossen.

Succes !

Vragenlijst finale computer hoger onderwijs

Praktische instructies

Deze instructies leggen uit hoe je aan je programma kan werken en hoe je het nadien moet indienen op de officiële server. We raden je aan dit te lezen en de inhoud tegelijk toe te passen op *Taak 0*.

Stap 1 – Open het skeletprogramma dat je moet vervolledigen

- **Open je persoonlijke map** door te dubbelklikken op het icoon *"Persoonlijke map van olymp-sup"* op het bureaublad.
- Die bevat een map genaamd `OI2011`. **Open ze**.
- De map `OI2011` bevat één map per probleem. **Open de map van de vraag** waaraan je wil werken.
- In de mappen van elk probleem, vind je opnieuw één map per programmeertaal. **Open de map die overeenkomt met de taal** waarin je gaat programmeren (je kan later altijd nog veranderen).
- In die map vind je een **bestand met broncode** genaamd `code.*` (de extensie hangt af van de programmeertaal), **een inputbestand** voor je programma (genaamd `input-example.txt`) en **een outputbestand** dat ermee overeenkomt (genaamd `output-expected.txt`). Het inputbestand bevat een voorbeeld van geldige input, waarvoor je programma als output de inhoud van het outputbestand moet teruggeven. Deze twee bestanden worden gebruikt om je programma te testen.
- Dubbelklik op het bestand met broncode om het te openen. Het programma *gedit* zal geopend worden om het bestand te lezen.

Stap 2 – Het compileren, uitvoeren en testen van je programma

- Wanneer je in *gedit* werkt, kan je **CTRL+R** typen om je programma te compileren, uit te voeren en te testen.
- Een console verschijnt in het onderste gedeelte van *gedit*, en toont je hoe de compilatie verloopt.
- Als je programma met succes werd gecompileerd, zal het worden uitgevoerd met als input het bestand `input-example.txt` dat in de map aanwezig is. Je mag dat bestand uiteraard ook aanpassen ! Het resultaat van je programma wordt weggeschreven naar het bestand `output.txt`.
- Als de uitvoering zonder problemen is verlopen, wordt het resultaat `output.txt` van jouw programma vergeleken met de verwachte output in het bestand `output-expected.txt`. Als je `input-example.txt` hebt aangepast, vergeet dan niet dat ook met `output-expected.txt` te doen.
- In alle gevallen, als een fout zich voordoet (bij compileren of uitvoeren) zal de informatie daarover verschijnen in de console van *gedit*.

Stap 3 – Je programma indienen

- **Open Firefox** (in het menu *« Toepassingen* > *Internet* > *Firefox »*).
- Typ de volgende URL in de adresbalk : `http://130.104.78.201`.
- Log in op deze website met de login en paswoord die zich op de eerste pagina van deze vragenlijst bevinden.
- Selecteer aan de rechterkant de vraag die je wilt indienen.
- Klik op *« submit a new solution »* om nieuwe code in te dienen. Als je wil vertrekken van wat je eerder had ingediend, selecteer dat dan en klik op *« See submitted source file »* om het te bewerken. In beide gevallen maak je zo een nieuwe submissie.
- Kies je programmeertaal (dit is noodzakelijk !). Daarna plak je je code in het daarvoor voorziene veld, of pas je aan wat er al in staat. Klik daarna op *« submit »*.

- Je indiening wordt nu verwerkt. Klik na enkele seconden op *« see the result »* om het resultaat te bekijken. Als het resultaat *« waiting »* is, wil dat zeggen dat jouw indiening nog in een wachtrij staat. Als er *« running »* verschijnt, wordt je code momenteel uitgevoerd. Wacht in beide gevallen enkele seconden (desnoods een minuut) voordat je op *« refresh »* klikt. Het is nutteloos om fanatiek de pagina te gaan verversen, dat vertraagt de website alleen maar voor alle deelnemers.
- Zodra je programma werd uitgevoerd, vind je op dezelfde pagina het resultaat voor alle tests. Voor elke test, wilt *« timeout »* zeggen dat je programma de toegelaten runtime op de server heeft overschreden. *« error »* wijst op een fout in de compilatie, uitvoering, of op het gebruik van functies die niet werden toegelaten. *« wrong output »* wijst erop dat de output van je programma niet gelijk is aan de verwachte output. Als je programma de uitvoeringstijd van een test overschrijdt (*« timeout »*), worden alle volgende tests niet meer uitgevoerd.

Hoe het systeem van indiening werkt

- Zodra je nieuwe code indient, wordt die in een wachtrij gezet. Ze zal zo snel mogelijk worden uitgevoerd maar dat hangt ook af van hoeveel anderen er recent nieuwe code hebben ingediend. Het is dus mogelijk dat aan het eind van de sessie de wachttijd voor je resultaten oploopt tot enkele minuten.
- Als je nieuwe code indient terwijl je op dezelfde vraag nog code in de wachtrij had staan, zal de oude code geannuleerd worden en wordt de nieuwe aan het eind van de wachtrij opnieuw toegevoegd.
- Als je na het einde van de proeftijd nog code in de wachtrij hebt zitten, zal die uiteraard nog meetellen. Je zal echter niet meer weten hoeveel punten ze je oplevert.
- **Alleen de best behaalde score op elke taak die je indient zal meetellen om je eindscore van het gedeelte aan de computer te berekenen.**
- Voordat je indient : zorg er zeker voor dat je alle tekstberichten die je in je programma laat uitprinten om te debuggen, ook terug verwijdert. Anders verschijnen ze op de standaard output, en wordt het resultaat van je programma incorrect !

Opmerkingen

- Documentatie voor elke taal bevindt zich in de map `OI2011/docs`.
- Als je hulp nodig hebt bij enkele dingen die op deze pagina staan, vraag het dan aan een toezichthouder.
- Als je je code nog niet hebt ingediend kan het altijd handig zijn om regelmatig ergens een kopie te bewaren, voor het geval dat je per ongeluk iets verkeerd doet in de editor. Deze *"backups"* kunnen veel gevloek voorkomen.
- De code in *gedit* zal nooit worden bekeken. Alleen de code die werd ingediend via de server telt mee voor je eindscore.

Je hebt slechts 2 uur de tijd voor deze proef. Verkies een tragere maar juiste oplossing boven een ambitieuze poging die uiteindelijk niet functioneert !

Taak 0 – Sum

Dit is een kleine oefening om je vertrouwd te maken met de werkomgeving en de server voor het indienen van de code. Je moet gewoon de som maken van twee positieve gehele getallen.

Taak

Schrijf een programma dat, gegeven 2 positieve gehele getallen, hun som berekent.

Limieten en beperkingen

Je programma hoeft enkel problemen te kunnen oplossen die zich binnen deze limieten bevinden. Alle testen die worden uitgevoerd zullen deze limieten respecteren.

- $1 \leq a, b \leq 100$, de getallen die opgeteld moeten worden ;

Wat er ook gebeurt, je programma zal stopgezet worden na **1 seconde** uitvoeringstijd. Je programma mag niet meer dan **10 MB** geheugen gebruiken.

Input

De input die je programma krijgt heeft het volgende formaat :

- De eerste en enige lijn bevat twee positieve gehele getallen a en b, gescheiden door één spatie.
- De input eindigt met een nieuwe lijn.

Output

Je programma schrijft de som van a en b naar de output, gevolgd door een nieuwe lijn.

Voorbeeld

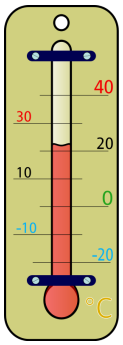
Gegeven de volgende input die aan je programma wordt gegeven :

```
10 81
```

De output van je programma moet dan zijn :

```
91
```

Taak 1 – Temperature

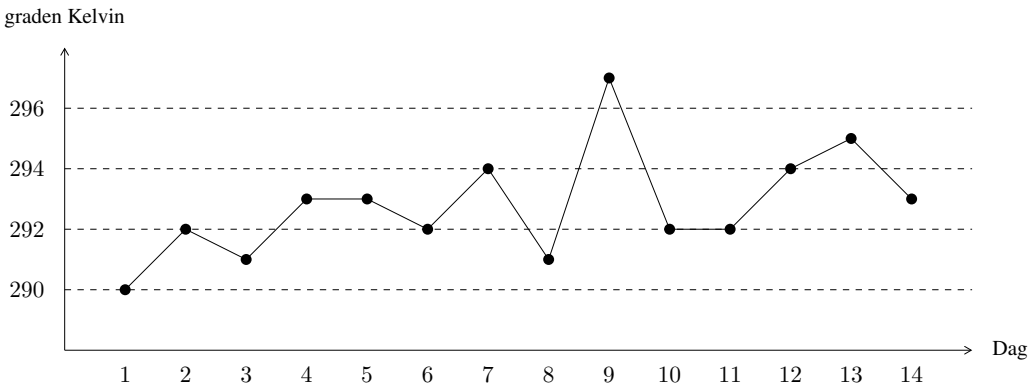


Je hebt een nieuwe baan als analyst bij het Koninklijk Observatorium van België. Die laatste heeft van de Algemene Directie Statistiek en Economische Informatie de vraag gekregen om de periodes te vinden waar de gemiddelde temperatuur het hoogst was.

Daarvoor krijg je een reeks gegevens die overeenkomen met metingen van de temperatuur, elke dag gedurende een zekere periode. Je moet verschillende dingen vinden :

- De hoogste temperatuur in die periode ;
- De hoogste gemiddelde temperatuur gedurende 7 opeenvolgende dagen ;
- De hoogste gemiddelde temperatuur gedurende 31 opeenvolgende dagen.

Om dit probleem op te lossen, denk je aan een algoritme dat veel algemener is dan dat. Jouw algoritme aanvaardt als parameter het aantal opeenvolgende dagen waarvoor je de maximale gemiddelde temperatuur wilt vinden. Dat aantal dagen is het *observatievenster*. Hier is een voorbeeld van temperatuurgegevens (gegeven in graden Kelvin) :



Nemen we een venster van grootte 1, dan is het maximale gemiddelde op dag 9, namelijk **297**. Nemen we venstergrootte 3, dan vinden we een maximaal temperatuurgemiddelde van dagen 12 tot en met 14 : $(294+295+293)/3 = \mathbf{294}$. Het is deze waarde die jouw programma moet vinden.

Taak

Schrijf een programma dat, gegeven een positief geheel getal F en een array tab van N positieve gehele getallen, het maximale gemiddelde teruggeeft dat we kunnen vinden onder alle sub-arrays van grootte F die we kunnen vormen in de array met gegevens tab.

Limieten

Je programma hoeft enkel problemen te kunnen oplossen die zich binnen deze limieten bevinden. Alle testen die worden uitgevoerd zullen deze limieten respecteren.

- $2 \leq F \leq 10\,000$;
- $F \leq N \leq 500\,000$;
- $0 \leq tab_i \leq 320$, de waarden in de array.

Wat er ook gebeurt, je programma zal stopgezet worden na **3 seconden** uitvoeringstijd. Je programma mag niet meer dan **256 MB** geheugen gebruiken.

Input

De input die je programma krijgt heeft het volgende formaat :

- De eerste lijn bevat twee positieve gehele getallen, gescheiden door één spatie : N en F ;
- De volgende N lijnen bevatten elk een positief geheel getal : een temperatuurmeting ;
- De input eindigt met een nieuwe lijn.

Output

De output die je moet produceren bevat één lijn met een positief geheel getal : het maximale gemiddelde dat je kan bekomen met een observatievenster van grootte F (afgerond naar beneden). Dat getal wordt gevolgd door een nieuwe lijn.

Voorbeeld

Gegeven de volgende input van 14 temperatuurmetingen, en een venster van grootte 3 :

```
14 3
290
292
291
293
293
292
294
291
297
292
292
294
295
293
```

Je programma moet dan aan de output schrijven (de oplossing is uniek !) :

```
294
```

Score

De totaalscore voor deze taak wordt berekend op het resultaat van twee groepen testen, die elk voor 50% van de punten meetellen.

- $2 \leq F \leq 25$ en $F \leq N \leq 1\,000$ **(5 tests)**
- $2 \leq F \leq 10\,000$ en $F \leq N \leq 500\,000$ **(5 tests)**

Elke test heeft slechts één enkele oplossing. Elk goed antwoord levert je dus punten op, een fout antwoord levert niets op.

Taak 2 – Repetition

Taak

Gegeven een reeks karakters van lengte N. Schrijf een programma dat de langste sub-reeks die herhaaldelijk voorkomt in deze reeks teruggeeft. Als er meerdere zulke langste reeksen zijn, geef dan de eerste ervan terug. De reeks karakters bevat hoofdletters en kleine letters, punctuatie (zintekens) en spaties. Hoofdletters en kleine letters worden als verschillende karakters beschouwd.

Bovendien mogen de herhalingen elkaar overlappen, bijvoorbeeld, de langste herhaalde sub-reeks in "lalala" is "lala".

Als voorbeeld, als de gegeven reeks "Lorem ipsum dolor a porttitor lorem ac ipsum tempor dolor" is, geeft je programma " ipsum ". Let op de twee spaties hierin, die eveneens herhaald worden.

Limieten en beperkingen

Je programma hoeft enkel problemen te kunnen oplossen die zich binnen deze limieten bevinden. Alle testen die worden uitgevoerd zullen deze limieten respecteren.

- $0 < N \leq 65\,530$
- De karakters in de reeks kunnen zijn : de 26 letters van het alfabet in hoofdletters en kleine letters, spaties, komma's, puntkomma's en punten.

Wat er ook gebeurt, je programma zal stopgezet worden na **3 seconden** uitvoeringstijd. Je programma mag niet meer dan **256 MB** geheugen gebruiken.

Input

De input die je programma krijgt heeft het volgende formaat :

- De eerste lijn bevat een positief geheel getal N : het aantal karakters in de karakterreeks die je gaat analyseren ;
- De tweede lijn bevat de reeks karakters om te analyseren ;
- De input eindigt met een nieuwe lijn.

Output

De output moet slechts 1 lijn bevatten, waarop de langste herhaaldelijk voorkomende sub-reeks van karakters staat. Deze sub-reeks wordt beëindigd met een nieuwe lijn.

Voorbeeld

Gegeven de volgende input die een reeks van 124 karakters bevat (merk op dat de volledige reeks karakters op één enkele lijn staat (de tweede)) :

```
124
Lorem ipsum dolor sit amet, consectetur adipisicing elit,
sed do eiusmod tempor incididunt ut labore et dolore magna
aliqua.
```

Je programma moet dan als output schrijven (de oplossing is uniek !) (let op de spatie aan het begin van de reeks) :

```
 dolor
```

Score

1. $0 < N \leq 1\,000$ **(5 tests)**
2. $20\,000 < N \leq 30\,000$ **(5 tests)**
3. $45\,000 < N \leq 50\,000$ **(5 tests)**
4. $60\,000 < N \leq 65\,530$ **(5 tests)**

Elke test heeft slechts één enkele oplossing. Elk goed antwoord levert je dus punten op, een fout antwoord levert niets op.