April 1, 2019

CS204 (Adv. Prog.) First Midterm Exam

1	2	3	4a	4b,c.d	5	6a	6b	7	TOTAL

Name and Lastname : Son Listesilici
SUNET User Name :

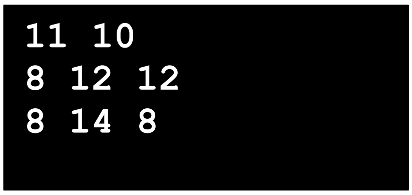
Notes: a) Please answer the questions only in the provided space after each question.
b) Duration is 100 minutes.
c) Closed-book, closed-notes, no calculators and computers. ½ (half) A4 size, single-sided, handwritten cheat-note page is allowed.
d) There must be seven pages (including this one) in this booklet. Please check it out!

QUESTIONS

1) (11 points) What is the output of the following program?

```
#include <iostream>
using namespace std;

void muslum (int * & a, int * b)
{
      *a=*a+2;
      b=a;
      a=new int;
      *a=8;
}

int * baba (int * valp)
{
      int * temp =  new int;
      *temp = *valp;
      *valp = *valp + 1;
      return temp;
}

int main()
{
      int val = 10;
      int * cs = baba(&val);
      cout << val << " " << *cs << endl;

      int * ks = cs;
      int * ps = ks;
      muslum (cs, ps);
      cout << *cs << " " << *ks << " " << *ps << endl;
      muslum (ps, cs);
      cout << *cs << " " << *ks << " " << *ps << endl;

      return 0;
}
```

Write your answer here



```
11 10
8 12 12
8 14 8
```

NAME:

2) (12 points) Consider the following *ronaldo.cpp* and *messi.h* files.

ronaldo.cpp

```
#include <iostream>
using namespace std;

#include "messi.h"

#define COMOLLI
#define MAHMUT 10

int main()
{
  halay = MAHMUT;

  #ifdef MESSI
    cout << COMOLLI "failed fener" << endl;
  #endif
  #ifndef RONALDO
    cout << halay+MAHMUT << endl;
  #endif
  #if MAHMUT - 10 == 1
    cout << "ver serceyi" << endl;
  #else
    cout << "teytey" << endl;
  #endif
  if (MAHMUT+halay != 2*MAHMUT)
    cout << "doomsday" << endl;
  else
    cout << "normal day" << endl;
  return 0;
}
```

messi.h

```
#ifndef MESSI
#define MESSI

  static int halay;

#endif
```

a) What is the <u>translation unit</u>? Fill in the box below.

```
#include <iostream>
using namespace std;
```

```
static int halay;

int main()
{
  halay = 10;

  cout << "failed fener" << endl;
  cout << halay+10 << endl;
  cout << "teytey" << endl;
  if (10+halay != 2*10)
    cout << "doomsday" << endl;
  else
    cout << "normal day" << endl;
  return 0;
}
```

b) Does this project compile and link (i.e. build) correctly? If not, specify the erroneous lines on the code and explain the error. If so, give the output of the program. In either case, write your answer in the box below.

It compiles and links correctly without any problem. The output is:

```
failed fener
20
teytey
normal day
```

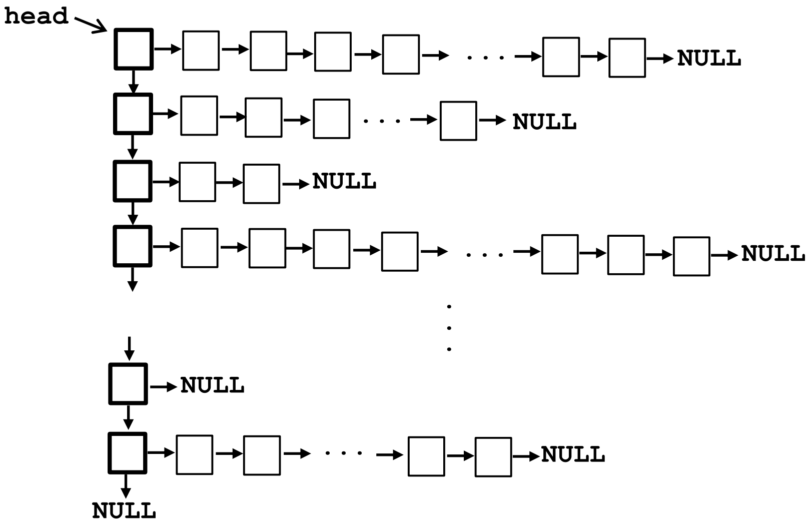
3) (18 points) Consider a 2D hybrid data structure as illustrated below. This data structure is made of two different node types. The first node of each row is of type `firstNode` and the rest is of type `regularNode`. Node structures are given below.

The `head` pointer points to first node of first row as shown in the figure. Each row of this structure is a linked list ending with `NULL` (by linking `right` pointers). Moreover, first nodes also create a vertical linked list ending with `NULL` (by linking `down` pointers).

There could be any number of rows. Each row starts with a `firstNode`, but there could be any number of `regularNode`s in each row (zero or more). This means, number of nodes in each row could be different. The figure below is for illustrative purposes only and does not imply any fixed number of nodes and/or rows.

```
struct regularNode
{
        int info;
        regularNode * right;
};
```

```
struct firstNode
{
        int info;
        regularNode * right;
        firstNode * down;
};
```



In this question, you may prefer not to attempt to solve it by signing the "not attempted" box below and secure 4 points. If you sign the "not attempted" box below, you accept that you did not answer this question and you will receive 4 points. In this case, your answer will not be graded even if you write something as solution.

<u>Not attempted</u>

Write a function that takes `head` as parameter and returns the sum of the info fields of all of the nodes (both `firstNode`s and `regularNode`s) of this data structure. As mentioned above, there could be any number of rows (zero or more). Each row starts with a `firstNode`, but there could be any number of `regularNode`s (zero or more).

```
int sum (firstNode * head)
{
    int total = 0;
    firstNode * vptr = head;
    while (vptr != NULL)
    {
         total = total + vptr->info;
         regularNode * rowptr = vptr->right;
         while (rowptr != NULL)
         {
              total = total + rowptr->info;
              rowptr = rowptr->right;
         }
         vptr = vptr->down;
    }
    return total;
}
```

NAME:

4)

a) (5 points) Consider the following macro definition.

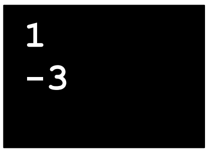
```
#define ZERO(N) N+N-2*N
```

What is the output of the following program piece? Show your work. If there is "division by zero" problem in any of them, please also specify this.

```
cout << ZERO(3)/ZERO(3) << endl;

cout << (ZERO(3))/ZERO(3) << endl;
```

3+3-2*3/3+3-2*3 = 3+3-6/3+3-6 = 3+3-2+3-6 = 1

(3+3-2*3)/3+3-2*3 = (3+3-6)/3+3-2*3 = 0/3+3-2*3 = 3-6 = -3



```
1
-3
```

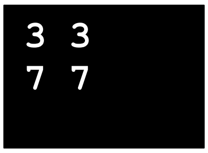
b) (2 points) Suppose you have 6 different .cpp files in a C++ Project that will be compiled separately, but will be linked together. In **at most** how many of them can you make the following global extern definition and use `myAverage` without any problem?

```
extern double myAverage;
```

We can have this extern definition in at most 5 cpp files since in the other one we have to have the actual global definition.

c) (4 points) What is the output of the following program piece?

```
int a=3, b=7;
int **p;
int *q;
q=&a;
p=&q;
cout << **p << " " << *q << endl;
a=b;
b=5;
cout << **p << " " << *q << endl;
```



```
3 3
7 7
```

d) (2 points) Fill in the blanks in the following sentence that explains the syntactic problem in the code line: `double * dp = calloc(1200, sizeof(double));`

Hint: the pointer types must be written in the blanks.

`calloc` returns ___**void**_____ pointer; it needs to be typecast to _____**double**___ pointer before assigning to `dp`.

NAME:

5) (17 points) Consider a regular **linked list** of which the node structure is given below.

```
struct node
{
     int info;
     node * next;
};
```

Linked list implemented as a class and the only private data member is `node * head;`

Assuming that there are odd number of nodes in the list object, the following incomplete member function aims to return the info field of the node in the middle of the list. For examples, if there are 17 nodes in the list, the function should return the info field of the 9th node; if there is one node, it should return the info field of that node; if there are 3 nodes, the function returns the info field of the 2nd node.

Complete this member function by filling in the boxes with appropriate code. You are not allowed to delete or update anything. Moreover, you cannot add anything other than the code that you are going to write in the boxes

Since we assume that there are odd number of nodes in the list, definitely the list is non-empty, but there could be any odd number of nodes in it.

Remark: As you see in the partial code, we pass over the list only once. This is not the easiest way of doing this task, but it is the most efficient way. Think of using the local pointers (p1 and p2 in the function below) in a clever way to accomplish this task.

```
int linkedlist::middleNodeInfo ()
{
     node * p1;
     node * p2;

     p1 = head ;

     p2 = head ;

     while ( p2->next!=NULL )
     {
          p1= p1->next ;

          p2= p2->next->next ;
     }
     return p1->info ;
}
```

p1 and p2 could be used interchangeably.

In this question, you may prefer not to attempt to solve it by signing the "not attempted" box below and secure 4 points. If you sign the "not attempted" box below, you accept that you did not answer this question and you will receive 4 points. In this case, your answer will not be graded even if you write something as solution.

Not attempted

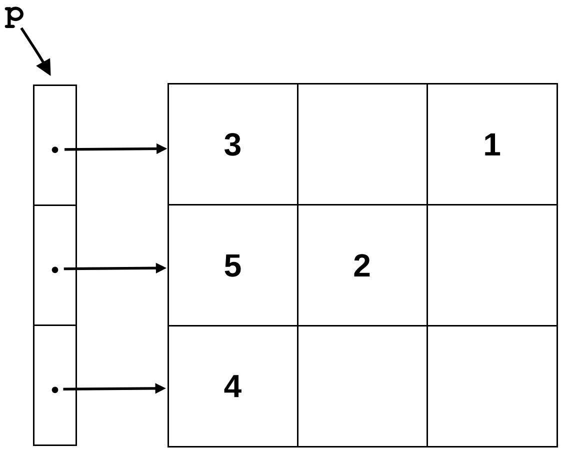
NAME:

6)

a) (5 points) Following piece of code creates and partially updates the content of a dynamic 2D array (i.e. dynamic matrix). Fill in the updated cells of the matrix in the figure below. Please only fill the cell that are updated by the code; others will remain blank.

Hint: there are only five cells updated and a particular cell is not updated twice.

```
int **p = new int * [3];
*p = new int [3];
p[1] = new int [3];
p[2] = new int [3];
**p = 3;
*p[2] = 4;
int *q = p[1];
*q++=5;
*q=2;
q = *p;
*(q+2)=1;
```



3		1
5	2	
4		

b) (9 points) Consider a **doubly linked list** of which the node structure is given below.

```
struct node
{
      string data;
      node * next;
      node * previous;
};
```

Write a function that takes a pointer to one of the nodes of a doubly linked list and returns the head pointer of the same list. The parameter pointer may point to any node in the list.

Linked list is **not** implemented as a class.

You may assume that the list is **not empty**. However, there could be any number of elements in it.

The function heading is given below. You are expected to write the function body.

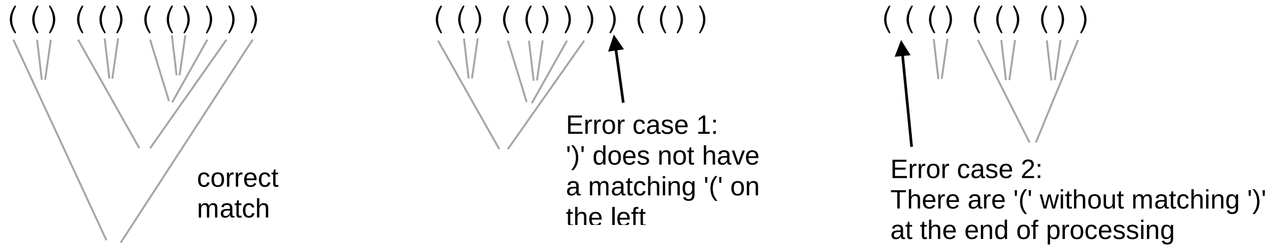
```
node * returnHead (node * anyNode)
{
   while (anyNode->previous != NULL)
   {
        anyNode = anyNode->previous;
   }
   return anyNode;
}
```

NAME:

7) (15 points) In this question, you will implement parenthesis matching rules over a string. Firstly, let us remember the parenthesis matching rule: each closed parenthesis ')' belongs to the nearest open parenthesis '(' on the left for which there is no closed parenthesis. Moreover, there are two possible error cases here:

(1) Having ')' without matching '(' This case may happen anywhere in the string.
(2) When the processing has finished, having remaining '(' without matching ')'

Correct match and the both error cases are exemplified below.



The function below is a partial solution for the following problem:

> Write a function that takes a string a parameter. You can assume that there are only `(` and `)` characters in the string. The function returns 0 if parenthesis matching rule holds over the parameter string. If there is a matching error, the function should return error code (1 or 2) mentioned above.

However, the function is incomplete. Complete this function by filling in the boxes with appropriate pieces of code.

You are not allowed to delete or update anything. Moreover, you cannot add anything other than the code that you are going to write in the boxes. Furthermore, you are not allowed to change the `DynIntStack` class. In other words, you can use only `DynIntStack` class' `push`, `pop`, and `isEmpty` member functions. Remember that `push` and `pop` are void functions and they take integer parameters. `isEmpty` does not take any parameters and returns Boolean.

```
int matchCheck (string expr)
{
    DynIntStack stack;
    int dummy;
    for (int i=0; i < expr.length(); i++)
    {
        if ( expr[i] == '(' )
            stack. push(i) ;
        else
        {
            if ( stack.isEmpty() )
                return 1;
            else
                stack. pop(dummy) ;
        }
    }
    if ( stack.isEmpty() )

        return 0;
    else
        return 2;
}
```

Pushing any integer is OK

In this question, you may prefer not to attempt to solve it by signing the "not attempted" box below and secure 3 points. If you sign the "not attempted" box below, you accept that you did not answer this question and you will receive 3 points. In this case, your answer will not be graded even if you write something as solution.

Not attempted