BITS PILANI, DUBAI CAMPUS
SECOND SEMESTER 2022 – 2023
THIRD YEAR CS
COMPREHENSIVE EXAMINATION (CLOSED BOOK)

A

Course Code: CS F363 | Date: 29.05.2023
Course Title: Compiler Construction | Max Marks: 40
Time: 12:30-3:30PM | Weightage: 20%

Answer PART A and PART B in separate answer books
Answer ALL questions
Pl. state any assumptions.

A1 Draw the symbol table using linked list for the given program **4M**

```
class MyClass1 {
  int mc1v1[2][4];
  float mc1v2;
  MyClass2 mc1v3[3];
  int mc1f1(int p1, MyClass2 p2[3]) {
    MyClass2 fv1[3];
    …
  }
  int f2(MyClass1 f2p1[3]) {
    int mc1v1;
    …
  }
}
class MyClass2 {
  int mc1v1[2][4];
  float fp1;
  MyClass2 m2[3];
  …
}
program {
  int m1;
  float[3][2] m2;
  MyClass2[2] m3;
  ...
}
float f1(int fp1[2][2], float fp2) {
  MyClass1[3] fv1;
  int fv2;
  …
}
int f2() {
...
}
```

A2 Construct the LL(1) Parsing table for the given grammar and check whether the grammar satisfies all the properties of LL(1) grammar. If not, state which property is not satisfied. **10M**

$S \rightarrow \mathbf{a}SA\mathbf{b} \mid \mathbf{b}SB\mathbf{c}$

$A \rightarrow A+B \mid \varepsilon$
$B \rightarrow *BC \mid \varepsilon$
$C \rightarrow \mathbf{a}C \mid \mathbf{a}B \mid \mathbf{d}$

A3 Consider the syntax directed definition given by the following grammar and semantic rules. Here N, I, F and B are non-terminals. N is the starting non-terminal, and #,0 and 1 are lexical tokens corresponding to input letters "#", "0" and "1", respectively. $X.val$ denotes the synthesized attribute (a numeric value) associated with a non-terminal X. I_1 and F_1 denote occurrences of I and F on the right-hand side of a production, respectively. For the tokens 0 and 1, $0.val$=0 and $1.val$=1. **8M**

$N \rightarrow I\#F$	$N.val=I.val + F.val$
$I \rightarrow I_1B$	$I.val=(2I_1.val) + B.val$
$I \rightarrow B$	$I.val=B.val$
$F \rightarrow BF_1$	$F.val=1/2(B.val+F_1.val)$
$F \rightarrow B$	$F.val=1/2\ B.val$
$B \rightarrow \mathbf{0}$	$B.val=\mathbf{0.val}$
$B \rightarrow \mathbf{1}$	$B.val=\mathbf{1.val}$

Using the CFG and the semantic rules given above and compute the value of the given input string **10#011** and show the calculation using annotated parse tree.

A4 Write a three-address code for the following program and represent the same using quadruples format. **10M**

```
for (int i = 0; i < 2; i++)
  {
    for (int j = 0; j < 3; j++)
    {
      for (int k = 0; k < 2; k++)
      {
        test[i][j][k];
      }
    }
  }
```

A5 Optimize the given code by identifying the suitable optimization techniques. Name the suitable code optimization technique(s) and write the optimized code. Assume all the variables are of integer data type. **4M**

```
int calculate(int x, int y) {
    int a = x + y;
    int b = x - y;
    int c = a * b;
    int d = c + a;
    int e = d / 2;
    int f = e * 3;

    return f;
}
```

A6 Consider the Grammar **4M**

$S \rightarrow A\mathbf{c}B$

$A \rightarrow \mathbf{a}A\mathbf{b} \mid \varepsilon$

$B \rightarrow \mathbf{a}B\mathbf{b} \mid \mathbf{c}$

Derive the rightmost sentential form of the given input string **abcacb** and find the corresponding handles at each step for reduction in bottom-up parsing.

BITS PILANI, DUBAI CAMPUS
SECOND SEMESTER 2022 – 2023
THIRD YEAR CS
COMPREHENSIVE EXAMINATION (CLOSED BOOK)

B

Course Code: CS F363 Date: 29.05.2023
Course Title: Compiler Construction Max Marks: 40
Time: 12:30-3:30PM Weightage: 20%

Answer PART A and PART B in separate answer books
Answer ALL questions
Pl. state any assumptions.

B1 Consider the following C program: **3M**

```
#include <stdio.h>
void main ()
{
  void e (int xx, int *nn);
  int x[5], i;
  n = 2;
  for (i = 0; i < 5; i += 1)
    {
      x[i] =  n;
      e (x[i], &n);
      n = n * 2 + 1;
    }
}
void e (int xx, int nn)
{
  int m, z;
  m = *nn * 2;
  z = xx + 3;

  printf (" m = %d z= %s \n", m, z);
}
```

The above program should display the following output:

```
m = 4 z= 5
m = 10 z= 8
m = 22 z= 14
m = 46 z= 26
m = 94 z= 50
```

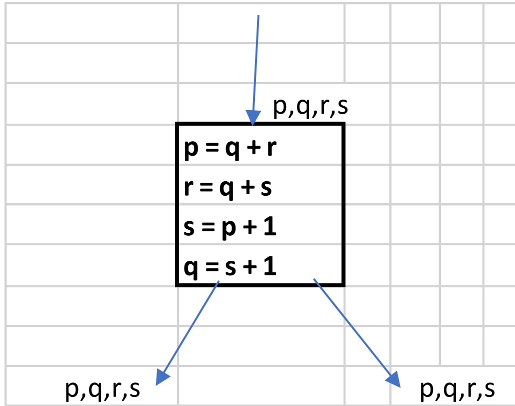
a) Find out syntax errors, semantic errors if any in the above program. Give reasons.
b) If there are any syntax errors or semantic errors in the above code, remove the errors and **Write down** the *corrected version of the program*.

B2 For the 'C' expression given below show the output of the following phases of a typical compiler. **i)** lexical analysis, **ii)** syntax analysis **iii)** semantic analysis. Also show the contents of the symbol table. **3M**

x = (p * q) / (r + s)

Assume the following declaration for identifiers (**float** p, s, x; **int** q, r), standard precedence for operators and suitable names for tokens.

B3 Consider the following basic block that is part of an inner loop. Compute the usage counts for the variables **p,q,r,s**. **4M**



B4 Explain the Generational Garbage Collection Process in JAVA with suitable diagrams. **8M**

B5 Consider the following three address code (TAC) instructions which constitutes a *basic block.* Assume **p, q, r** and **s** are program variables that are live on exit from the block. **TABULATE** the **liveness** and **next-use** information for the variables in each of the TAC instruction. **5M**

Instruction							Symbol Table								
	Liveness			Next-use				Liveness				Next-use			
	src	**op1**	**op2**	**src**	**op1**	**op2**		**p**	**q**	**r**	**s**	**p**	**q**	**r**	**s**
1) p=q-r							1) p=q-r								
2) s=q+r							2) s=q+r								
3) r=p+s							3) r=p+s								
4) q=p-s							4) q=p-s								
							Initial	1	1	1	1	N	N	N	N

B6 Break the following Intermediate Code (3AC) into BASIC BLOCKS and Write the statements of each Basic Block. **6M**

```
1)  i=1
2)  j=1
3)  t1 = 10 * i
4)  t2 = t1 + j
5)  t3 = 8 * t2
6)  t4 = t3 - 88
7)  a[t4] = 0.0
8)  j = j + 1
9)  if j <= goto (3)
10) i = i + 1
11) if i <= 10 goto (2)
12) i = 1
13) t5 = i - 1
14) t6 = 88 * t5
15) a[t6] = 1.0
16) i = i + 1
17) if i <= 10 goto (13)
```

B7 *Design of a Simple Code Generator:* Consider the following three-address code (TAC) statements which forms a basic block. **8M**

```
t = a + b
u = a + c
v = t - u
a = d
d = v - u
```

Assume that the program variables **a**, **b**, **c** and **d** are live on exit from the block but not those temporary ones <*t, u, v*>. Assume that your machine has 3 registers R1, R2, and R3. The <*register* and *address descriptors*> are appropriately initialized, as given in the table below.

a) **Generate** the target code for each of the TAC instructions using the *simple code generation algorithm*.

b) **Tabulate** the contents of the *register descriptor* and the *address descriptor* after each TAC instruction in the below format.

	R1	R2	R3	a	b	c	d	t	u	v
Initial Configuration				a	b	c	d			

TAC Statement	Machine Code	R1	R2	R3	a	b	c	d	t	u	v
….	….	….	….	….	….	….	….	….	….	….	….

B8 Consider the following LEX program. **3M**

```
dot .
Hello Hello
%%
{dot} printf("Hi\n");
{Hello} printf("Hello\n");
%%
```

Write the output of the above LEX program for the following inputs. Justify your answer in a sentence.

Inputs

a) Hi

b) Hello

c) Hello World