Assignment 1 of CE2004, Principles of Programming Languages

Score: 100 points

Due Time: 14:00 16th April

P.S.:

(1) You need to type your answers in a file and print them out in answer sheets, then submit your answer sheets to the TAs.

(2) Late submission will not be accepted.

(3) You can discuss these questions with your classmates; however, copying other student's answers is strictly prohibited.

==

(1) (4 points)

Compiler Optimization may cause what results in an executable file?

Ans.

(2) (12 points)

(a) What follows is a C program.

```
# include <stdio.h>
int   a;

int bar(int x, int y)
{ int  b;

  return b = x+y;
}
int main()
{ int *p;

  p = (int *) malloc (sizeof(int));
  *p = bar (8,9);
}
```

In the above program, (i) which variables are static variables? (ii) And which variables are stack dynamic variables? (iii) And which variables are explicit-heap dynamic variables?

P.S.: A function formal parameter is also deemed as a variable.

(b) What follows is a Java program excerpt.

```
class Circle
{
  int setVariable(int s)
  {  int r;

      r=6;
      return s+r;
  }
}
public class ShowArea
{
   public static void main(String args[])
   {
      Circle cir= new Circle();
      int a;

      a= cir.setVariable(8);
   }
}
```

In the above program, (i) which variables are static variables? (ii) And which variables are stack dynamic variables? (iii) And which variables are explicit-heap dynamic variables?

Ans.

(3) (9 points)

What follows is an excerpt of a Javascript program. Assume before location 1, variable `list` has never been used.

```
        :                  -- location 1
     list = [1, 2]
     prefix= list         -- location 2
     prefix = 47
     list = prefix        -- location 3
        :
```

(a) At location 1, what is the data type of variable `list`?

(b) At location 2, what is the data type of variable `prefix`?

(c) At location 3, what is the data type of variable `list`?

Ans.

(4) (12 points)

A program consists of the following two files, `fileu.c` and `filev.c`

```
/*============ fileu.c ================*/
int a=100;        // location 1
extern int t;     //  location 2
int bar(int y)    // location  3
{int x;           // location  4
 x=y+t;             // location 5
 return(x);
}                   // location 6

/*============ filev.c ================*/
#include<stdio.h>
int  t=9;                 // location 7
extern int a;            // location 8
extern int bar(int);    // location 9
int main()               // location 10
{ int z;                 // location 11
  printf("a=%d\n",a);
  printf("bar(3)=%d\n",bar(3));
}                                 //location 12
```

(a) List the locations of all variable definitions in the above two files.

(b) List the locations of all variable declarations in the above two files.

(c) List the locations of all function definitions in the above two files.

(d) List the locations of all function declarations in the above two files.

P.S.: A function formal parameter is also deemed as a variable.

Ans.

(5) (8 points) What follows is a C program.

```
#include <stdio.h>
int total_income, total_visitors_global;

void zoo(char *name, int visitors)
{int adult, children;
 static  int total_visitors=0;
     :
 total_visitors=total_visitors+visitors;   // location 1
 total_visitors_global=total_visitors;
      :
}
int main()
{
  int ticket_price_each_animal_type=2;

  printf("Good Morning!\n");                // location 2
  zoo("giraff", 600);
  zoo("elephant", 300);
  zoo("hippo",100);
  total_income=ticket_price_each_animal_type*total_visitors_global;
            :
}
```

(a) At location 1, list the names of variables or parameters that have memory assigned to it.

(b) At location 2, list the names of variables or parameters that have memory assigned to it.

Ans:

(6) (8 points)

Assume each integer variable uses four bytes to store it values. And each float point variable uses four bytes to store its value. For the following two `C` program excerpts, (a) and (b), which of them have a type error? Explain your answers.

(a)

```
 int a;
 union course
{
  int    b;
  float   c;
} security;
security.b = 3;    // location 1
a = security.b;    // location 2
```

(b)

```
int a;
union course
{
  int    b;
  float   c;
} security;
security.c = 3.3;  // location 3
a = security.c;    //  location 4
```

Ans.

(7) (6 points)

What follows is the content of program `add_a.c`.

```
/*----------------------------------------------------*/
#include <stdio.h>
int a=1 , b=6;
int c[10000]={1};
int main()
{
  a=b+c[0];    /* location 1*/
}
/*----------------------------------------------------*/
```

Assume `add_a.exe` is the executable of `add_a.c`.

What follows is the content of program `add_b.c`.

```
/*-------------------------------------------------------*/
#include <stdio.h>
int a=1 , b=6;
int c[10000];
int main()
{
  c[0]=1;
  a=b+c[0];      /*location 2*/
}
/*--------------------------------------------------------*/
```

Assume `add_b.exe` is the executable of `add_b.c`.

(a) At location 1 of `add_a.c` what is the value of variable `a`?

(b) At location 2 of `add_b.c` what is the value of variable `a`?

(c) For files `add_a.exe` and `add_b.exe`, which of these two files has larger size and why?

Ans.

(8) (12 points)

```
#include <stdio.h>
int a;
int b=1;
void candy()
{ int c;
   c=100;
}
void bar()
{ int d;
  static  int e;

  if(a==3)
    e=b;
  else
    candy();
  a=2;
}
```

```
main()
{ int g;

  a=3;       //location 1
  bar();     //location 2
  g=100*b;   //location 3
  bar();     //location 4
  g=200+a;   //location 5
}
```

(a) For the above program, when the statement at location 1 is executed, how many variables, including static variables and stack-dynamic variables, have been created?

(b) For the above program, when the statement at location 3 is executed, how many variables, including static variables and stack-dynamic variables, have been created?

(c) For the above program, when the statement at location 5 is executed, how many variables, including static variables and stack-dynamic variables, have been created?

Ans.

(9) (9 points)

Assume INTEGER and REAL are special words used to define the data types of variables in a language. Notation `;’ is used to define the end of a statement.

(a) What are ``special words,” ``key words,” and ``reserved words?”

(b) If INTEGER and REAL are keywords in a language, then are the following statements correct?

```
INTEGER REAL, INTEGER_A;
REAL INTEGER, REAL_A;
```

(c) If notation $+_i$ is the operator used to add to two integer numbers. In other words, the only legal operand type of $+_i$ is type INTEGER. And $+_f$ is the operator used to add two real numbers. In other words, the only legal operand type of $+_f$ is type REAL. And the language is a strongly typed language. Which of the following statements are correct?

(i) REAL = REAL $+_i$ INTEGER;
(ii) INETGER = REAL $+_f$ REAL_A;
(iii) REAL = REAL $+_i$ INTEGER_A;
(iv) INTEGER = INTEGER $+_f$ REAL_A;

Ans.

(10) (10 points)

Good language readability can improve writability.

Good language writability is detrimental to readability.

(a) Which one of the above two statements is correct? Which one of the above two statements is wrong?

(b) Give your explanation.

Ans.

(11) (10 points)

```
#include <stdio.h>
int a;
int b=1;

void candy()
{ int c;

  c=100;
}

void bar(int p)
{int d;
  static  int e;
      :
}
main()
{ int g;

  bar(3);
  g=100*b;    //location 1
      :
}
```

(a) For variables `a`, `b`, `c`, `d`, `e`, and `g` in the above program, list the segments (e.g. data segment, stack segment, or BSS segment) that provide storage for the variables in their lifetime.

(b) There are four variable categories, static, stack-dynamic, explicit heap-dynamic, implicit heap-dynamic. After the above program is executed, at location 1, which categories of variables have been created?

Ans.