CS 101 (Autumn 2024)
Mid-Semester Exam

15 September 2024

Instructor: Manoj Prabhakaran

Roll Number	SAMPLE
Division and Group	
Name	

Q.No.	**Marks / max**	**Graded By**	**Verified By**	**Re-grading** (if requested)
1	/ 6			
2	/ 3			
3	/ 5			
4	/ 6			
5	/ 6			
6	/ 8			
7	/ 6			
TOTAL	/ 40			

Please read the following instructions carefully before you start.

- Write your roll number, name, and group number in the space provided. A paper without a roll number and name will NOT be graded.
- Write your answers neatly with a blue/black pen on this question paper itself in the space provided for each question. At the end, you must submit this paper to the invigilator.
- Rough pages will NOT be provided. Use the empty space in the margins.
- Please note that your answers should NOT include any programming concept that hasn't been covered in the class so far. If such answers are found, they shall NOT be graded.
- No clarifications will be provided on any questions. When in doubt, make suitable assumptions, state them clearly, and proceed to solve the problem. If your answer depends on any assumption you have made, the assumption must be explicitly stated in your paper.
- In some questions, you are provided code snippets. Assume that the code snippet is enclosed suitably within the main, correct header files are included, etc., and therefore, the code compiles.
- All the best!

Translation

- शुरू करने से पहले कृपया निम्नलिखित निर्देशों को ध्यान से पढ़ें।
- दिए गए स्थान पर अपना रोल नंबर, नाम और ग्रुप नंबर लिखें। बिना रोल नंबर और नाम के पेपर को grade नहीं दिया जाएगा।
- इस प्रश्नपत्र पर ही प्रत्येक प्रश्न के लिए दिए गए स्थान पर अपने उत्तर नीले/काले पेन से साफ-सुथरा लिखें। अंत में आपको यह पेपर निरीक्षक के पास जमा करना होगा।
- रफ पेज उपलब्ध नहीं कराए जाएंगे. margin में दिए खाली जगह का उपयोग करें.
- कृपया ध्यान दें कि आपके उत्तरों में कोई भी प्रोग्रामिंग अवधारणा शामिल नहीं होनी चाहिए जिसे अब तक कक्षा में शामिल नहीं किया गया है। यदि ऐसे उत्तर पाए जाते हैं, तो उन्हें grade नहीं किया जाएगा।
- किसी भी प्रश्न पर कोई स्पष्टीकरण नहीं दिया जाएगा. जब संदेह हो, तो उपयुक्त धारणाएँ बनाएं, उन्हें स्पष्ट रूप से बताएं और समस्या को हल करने के लिए आगे बढ़ें। यदि आपका उत्तर आपके द्वारा की गई किसी धारणा पर निर्भर करता है, तो धारणा को आपके paper में स्पष्ट रूप से बताया जाना चाहिए
- कुछ प्रश्नों में आपको code snippet दिए गए हैं। मान लें कि code snippet main program के अंदर लिखा है, सही header files include की गयी हैं, आदि, और इसलिए, code compile होता है।
- शुभकामनाएं!

Question 1 [6 marks]

1. What is the output produced by the following code?

   निचे दिए गए program का output क्या होगा?

```
#include <iostream>
int main() {
  int a = 0, b = 0;
  std::cout << (a>b ? a-1 : b+1)
            << std::endl;
}
```

```
1
// 2 marks
// 1 mark for answering -1
```

2. Consider the following statements.

   निचे दिए गए statements पर विचार करे

```
int a = 3.0, b = 2.0;
a / b;
```

   What is the type and value of the expression in the second line?

   दूसरी पंक्ति में expression का type and value क्या है?

Type of `a/b` : `int // 1 mark`

Value of `a/b` : `1 //1 mark`

3. Match the two statements (i) and (ii) below, with equivalent ones from the statements (A) - (C).

   नीचे दिए गए दो statements (i) और (ii) को equivalent statements (A) - (C) के साथ मिलाएँ.

(i) `if( a>10 && b>10) f();`

(ii) `if( a>10 || b>10) f();`

(A) `if (a>10) f(); else if (b>10) f();`

(B) `if (a>10) f(); if (b>10) f();`

(C) `if (a>10) if (b>10) f();`

(i) is equivalent to `(C) // 1 Mark`

(ii) is equivalent to `(A) // 1 Mark`

Question 2 [3 marks]

Rough work

```
///////////////////////////////////
#include <iostream>
int main() {
  int i;
  for (i=0; i<3; i++) {
    int i = 0;
    std::cout << "i = " << i << std::endl;
  }
  std::cout << "i = " << i << std::endl;
}
///////////////////////////////////
```

Consider the program given above. In the box given below, write down its output.

ऊपर दिए गए program पर विचार करें। नीचे दिए गए box में इसका output लिखें।

```
If i = 0 is mentioned once
or twice, then 1 mark, and if
thrice, then 2 marks
i = 0
i = 0
i = 0
i = 3// 1 mark
```

Question 3 [5 marks]

```
///////////////////////////////////
#include <iostream>
using std::cin; using std::cout;
struct Pt {double a,b; };
void addPt(Pt& p, const Pt& q) {
    p.a += q.a; p.b += q.b;
}
void readPt(Pt& p) {
    cin >> p.a >> p.b;
}

void writePt(const Pt& p) {
    cout << p.a << " " << p.b << std::endl;
}

int main() {
  Pt p,q;
  readPt(p); readPt(q);
  cout << "Line 1: "; writePt(p);
  addPt(p,q);
  addPt(q,p);
  cout << "Line 2: "; writePt(p);
  cout << "Line 3: "; writePt(q);
}
///////////////////////////////////
```

Consider the program given above. The first two lines of output from the program (for some input) are shown in the box below. Fill in the remaining output.

उपर दिए गए program पर विचार करें। Program से प्राप्त पहले दो output लाइनों (कुछ input के लिए) को नीचे box में दिखाया गया है। बाकी output भरें।

```
Line 1: 3 4

Line 2: 6 6

Line 3: 9 8
// 5 marks if all correct
// 2.5 marks for any of the
following:
9 *
* 8
8 9
3 2 (original q)
6 6 (adding original p to q)
```

Rough work

Question 4 [6 marks]

```
//////////////////////////////////////////
#include <iostream>

TYPE1 adjust (TYPE2  x, TYPE3 y) {
  x = (x/y)*y;
}

int main() {
  int x = 10;
  adjust(x,4);
  std::cout << x << std::endl;
}
/////////////////////////////////////////
```

The above program has a function `adjust` which is missing the type specifiers (indicated as TYPE1, TYPE2, TYPE3).

उपर दिए गए program में एक function `adjust` है जिसमें type specifiers (जिन्हें TYPE1, TYPE2, TYPE3 के रूप में दर्शाया गया है) गायब हैं।

Suppose the program compiles without errors when three `#define` directives are added at the top, and further, when it is executed, produces the following output:

मान लीजिए कि program बिना किसी errors के compile होता है जब शीर्ष पर तीन `#define` directives लिखे जाते हैं, और इसके बाद, जब इसे execute किया जाता है, तो निचे दिया गया output देता है:

```
8
```

In the box provided, fill in the three directives mentioned above, which will complete the program. Justify each of your answers.

दिए गए box में, तीन directives को भरें जो उपर दिए गए program को पूरा करेंगे। प्रत्येक उत्तर को उचित ठहराएँ।

1 mark each for correct type and 1 mark each for correct justification

#define TYPE1 ____void____

#define TYPE2 ____int&____

#define TYPE3 ____int____

Alternative: const int&

Alternatives to int: char, short, long, long long. Optionally unsigned.

Justify your answer for TYPE1:
the function has no return statement.

Justify your answer for TYPE2:
calling the function with x as first argument results in x being changed. Since x is int, int& is the only possibility.

Justify your answer for TYPE3:
y must be an integer type (int, optionally unsigned/short/long etc.), rather than a floating point type, since with x=10, y=4, (x/y)*y is not equal to x. It is not a reference, since the function is called with a constant (rvalue) rather than a variable (lvalue). But can be a const reference (rvalues can be passed as const references).

Rough work

Question 5 [6 marks]

In this problem, we consider raising a number x to a (non-negative) integer power n, to compute x^n, using the idea of *repeated squaring* as follows.

Consider computing x^8. One can compute it as $((x^2)^2)^2$, by squaring just three times. In fact, along the way, we compute x, x^2, x^4, x^8. Now suppose we wish to compute x^{13}. Since $13 = 8+4+1$ this can be accomplished as $x^{13} = x^8x^4x^1$, where each of x^1, x^4 and x^8 is computed along the way to computing x^8. This idea can be extended to work for any n.

Consider the function below, which implements this approach. There are 4 expressions which are indicated as EXP1 through EXP4. Fill in what these expressions should be, so that the function correctly implements the above algorithm.

इस problem में, हम एक number x को एक (non-negative) integer power n raise करने का विचार कर रहे हैं, ताकि x^n को compute किया जा सके, और इसके लिए *repeated squaring* का उपयोग करते हैं।

मान लीजिए कि हमें x^8 को compute करना है। इसे $((x^2)^2)^2$ के रूप में केवल तीन बार square करके निकाला जा सकता है। वास्तव में, इस प्रक्रिया में हम x, x^2, x^4, x^8 को compute करते हैं। अब मान लीजिए कि हमें x^{13} को compute करना है। चूंकि $13 = 8 + 4 + 1$, इसे $x^{13} = x^8x^4x^1$ के रूप में निकाला जा सकता है, जहाँ x^1, x^4 और x^8 को compute करते हैं x^8 और compute करने दौरान की जाती है। यह विचार किसी भी n के लिए काम करने के लिए बढ़ाया जा सकता है।

नीचे दिए गए function पर विचार करें, जो इस approach को implement करता है। इसमें 4 expressions दिए गए हैं, जिन्हें EXP1 से EXP4 कहा गया है। आपको यह बताना है कि ये expressions क्या होने चाहिए, ताकि function इस algorithm को सही ढंग से implement कर सके।

```
/////////////////////////////////////////
double raise(double x, unsigned int n) {
  double out;
  for(out = 1; n > 0; n = EXP1) {
    if (EXP2)
      out = EXP3;
    x = EXP4;
  }
  return out;
}
/////////////////////////////////////////
```

Hint 1: Writing n as the sum of powers of 2 (e.g., $13 = 8 + 4 + 1$) involves the same computation as transforming n into its binary representation.

Hint 2: Try out your solution with small values of n like 0, 1, 2, 3.

संकेत 1: n को 2 की sum of powers के रूप में लिखना (जैसे $13 = 8 + 4 + 1$) वही computation है जो n को उसके binary representation में बदलने पर होती है।

संकेत 2: अपना solution को n के छोटे values जैसे 0, 1, 2, 3 के साथ आजमाएं

1.5 marks each

EXP1: n / 2

EXP2: n % 2 != 0 (or simply n%2)

EXP3: out * x

EXP4: x * x

Rough work

Question 6 [8 marks]

In each of the following pieces of code, there is a significant error that the compiler will complain about. (These are not typographical errors.) **Describe the error** and suggest a small edit to **fix it**, so that the program produces the desired output.

You should only change parts directly causing the error.

नीचे दिए गए code के हर हिस्से में एक महत्वपूर्ण गलती है जिसकी शिकायत compiler करेगा। (ये typographical गलती नहीं हैं।) **गलती का वर्णन करें** और एक छोटा सा बदलाव **सुझाएं** ताकि program desired output दे सके।

आपको केवल उन भागों को बदलना है जो सीधे गलती का कारण हैं।

1. [2 marks]

```
1 #include<iostream>
2 int main() {
3   int x = 3;
4   int& y;
5   y = x;
6   y++;
7   std::cout << x << std::endl;
8 }
```

Desired output: 4

1 mark for describing the error and 1 mark for correctly fixing it

Error: A reference variable needs to be initialized when declaring it. In line 4 it is declared without initializing.

Fix:
Replace lines 4 and 5 with:
int& y = x;

2. [3 marks]

```
1 #include<iostream>
2 void f(char x) {
3   x++;
4 }
5 int main() {
6   int x = 3;
7   x = f(x);
8   std::cout << x << std::endl;
9 }
```

Desired output: 4

1 mark for describing the error and 2 marks for fixing it

Error:
The function f is declared as void and does not return anything. However, in the main, the result of function call 'f(x)' is assigned to variable 'x'.

Fix:

Method 1

Line 2: int f(char x) {
Line 3: x++; return x;
Return type can be int, char or any numerical type (e.g., short, float).
-1 mark if input type changed (unnecessarily) to int or int& but no other error;
-1.5 marks for missing one of the two edits.

Method 2

Line 2: void f(int& x) {
Line 7: f(x);
-1 marks for using char& instead of int&.
-1.5 for missing one of the two edits.

3. [3 marks]

Rough work

```
#include<iostream>
int main() {
  int x = 3;
  (x++)++;
  std::cout << x << ", ";
  --(--x);
  std::cout << x << std::endl;
}
```

Desired output: `5, 3`

1.5 marks for correctly describing the error and 1.5 marks for fixing it

Error: The operation EXP++ requires EXP to have a memory location (i.e., it should be an "lvalue"). However, x++ is a temporary variable that doesn't have a memory location associated with it (it is only an "rvalue"). Hence (x++)++ is illegal.

Fix:
```
(++x)++; OR
++(++x); OR
x = x + 2; OR
x += 2;
```

Rough work

Question 7 [6 marks]

In each of the following problems, choose the correct answer(s) and justify. **Correct answers without correct justifications will not get any marks.**

नीचे दिए गए प्रत्येक problems में, सही उत्तर चुनें और इसका सही कारण बताएं। **सही उत्तर बिना सही कारण के अंक नहीं प्राप्त करेंगे।**

1. **Bitwise operations** [3 marks]

   Among the following expressions, one converts the `char` type variable `x`, if it is an uppercase or a lowercase letter, into lowercase (e.g., if `x` is `'B'` or `'b'`, it will become `'b'`), and another one converts it into uppercase. Identify those two expressions (say which is which) and justify your answer.

   Hint: You may recall that in ASCII code, the uppercase letters occupy the range 65 to 90, and the lowercase letters occupy the range 97 to 122. Corresponding uppercase and lowercase letters differ in a specific bit position.

   - ☐ `x |= 'A' ^ 'a'` to lowercase
   - ☐ `x &= 'A' ^ 'a'`
   - ☐ `x |= ~('A' ^ 'a')`
   - ☐ `x &= ~('A' ^ 'a')` to uppercase

   For each of the two questions (to uppercase, to lowercase), 1 mark for choosing the correct option, with at least a partial justification (no marks if no justification; also no marks if a wrong choice is also selected); additionally, 0.5 mark if the justification is fully correct.

   Justification:

   For any letter in the alphabet, the bit representations of the lowercase and uppercase ASCII characters differ in exactly one position (as mentioned in the hint), with the uppercase character having 0 in that position and the lowercase character having 1.

   ('A' ^'a') has 1 in that position and 0 elsewhere; its complement has 0 in that position and 1 elsewhere.

   Bitwise OR with ('A' ^'a') sets this bit to 1, making the character lowercase.

   Bitwise AND with ~('A' ^ 'a') sets this bit to 0, making the character uppercase.

   Further optional elaboration:
   'A' (ASCII 65) is 01000001 in binary.
   'a' (ASCII 97) is 01100001 in binary.
   The only difference between the two is the 6th bit:
   In 'A': 6th bit is 0.
   In 'a': 6th bit is 1.
   So, 'A' ^'a' is 00100000 in binary (32 in decimal). ~('A' ^'a') is 11011111 in binary.

   नीचे दिए गए expressions में से एक `char` प्रकार के variable `x` को, यदि यह एक uppercase या lowercase अक्षर है, तो lowercase में बदलता है (जैसे, यदि `x` `'B'` या `'b'` है, तो यह `'b'` में बदल जाएगा), और दूसरा इसे uppercase में बदलता है। उन दोनों expressions की पहचान करें (बताएं कि कौन सी कौन सी है) और अपने उत्तर का उचित कारण बताएं।

   संकेत: आप याद कर सकते हैं कि ASCII code में, uppercase अक्षर 65 से 90 के बीच होते हैं, और lowerase अक्षर 97 से 122 के बीच होते हैं। संबंधित uppercase और lowercase अक्षर एक विशिष्ट bit position में differ करता है।

2. **Integer Range** [3 marks]

Which of the following expressions *can be* `true` for two `int` type variables `x` and `y`. Justify with examples.

Hint: Recall the circular arrangement of negative and non-negative values represented using an n-bit integer data type. Adding positive numbers correspond to moving clockwise, and adding negative numbers correspond to moving anti-clockwise. (You may use a smaller n to help with reasoning.)

नीचे दिए गए expressions में से कौन से `true` हो सकते हैं दो `int` प्रकार के variables `x` और `y` के लिए। उदाहरणों के साथ सही ठहराएं।

Negative और Non-negative values जो n-bit integer data type से represent की जाती है, उसके circular arrangement को याद करे। Positive number add करने पर clockwise movement दर्शाता है और non-negative values anti-clockwise movement दर्शाता है। (आप एक छोटी value n की मदद से reasoning कर सकते है)

- ☐ `x>0 && y>0 && (x+y)<0`
- ☐ `x<0 && y<0 && (x+y)>0`
- ☐ `x>0 && y>0 && (x+y)==0`
- ☐ `x<0 && y<0 && (x+y)==0`

Justification:

0.5 mark each for each of the 4 options correctly selected/unselected, with at least a partial justification. Additionally, 0.25 mark each for each of the justification being fully correct.

Answer: Options 1, 2, 4.

Justification:

Option 1 can be true: If x and y are both positive and large enough, their sum can exceed the maximum value an int can store, causing an overflow and wrapping around to a negative value.

Example: For n-bit int (typically $n = 32$), if a $= 2^{n-1} - 1$ (this is the maximum positive value representable as int) and b $= 1$, then a+b evaluates to -2^{n-1} (the largest absolute value of a negative number that is representable as int).

Option 2 can be true: If x and y are both negative, and their sum becomes more negative than an int can hold, the result will wrap to a positive value.

Example: If a $= -2^{n-1}$ (most negative value) and b $= -1$, then a+b evaluates to $2^{n-1} - 1$ (the most positive value).

Option 3 is always false: If x+y == 0 when both $x > 0, y > 0$ (i.e., x steps clockwise followed by y steps clockwise gets you back to the starting point), then it must be the case that x+y (as integers) equals 2^n. However, since x, y are positive integers that can be represented as int, $x < 2^{n-1}$ and $y < 2^{n-1}$, and hence their sum cannot be 2^n.

Option 4 can be true: This happens exactly for one choice: x $= -2^{n-1}$ and y $= -2^{n-1}$.

Rough work