

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN



BÁO CÁO BÀI TẬP 2

| Đề tài |

KHẢO SÁT SỐ THỰC

| Sinh viên thực hiện |

Nguyễn Thái Bảo

23120023

| Giáo viên hướng dẫn |

Thầy Lê Viết Long

Môn học: Hệ thống máy tính

Thành phố Hồ Chí Minh - 2025

MỤC LỤC

| | |
|------------------------------|---|
| MỤC LỤC | 2 |
| 1. THÔNG TIN SINH VIÊN | 3 |
| 2. ĐÁNH GIÁ..... | 4 |
| 3. KẾT QUẢ BÀI LÀM | 5 |
| BÀI 1..... | 5 |
| BÀI 2..... | 5 |
| BÀI 3..... | 6 |
| BÀI 4..... | 7 |

1. THÔNG TIN SINH VIÊN

Họ và tên: Nguyễn Thái Bảo

Mã số sinh viên: 23120023

Lớp: 23CTT1

Email: 23120023@student.hcmus.edu.vn

2. ĐÁNH GIÁ

| Bài | Ghi chú | Đánh giá mức độ hoàn thành |
|-----|--|----------------------------|
| 1 | Hoàn thành đầy đủ yêu cầu: <ul style="list-style-type: none">- Viết chương trình nhập số chấm động- Xuất ra biểu diễn nhị phân từng thành phần của nó. | 100% |
| 2 | Hoàn thành đầy đủ yêu cầu: <ul style="list-style-type: none">- Viết chương trình nhập biểu diễn nhị phân của số chấm động (có hỗ trợ nhập khoảng trắng phân cách từng thành phần)- Xuất ra biểu diễn thập phân tương ứng | 100% |
| 3 | Hoàn thành đầy đủ yêu cầu: <ul style="list-style-type: none">- Dùng 2 hàm đã viết ở trên để khảo sát các câu hỏi- Phân tích câu hỏi, viết chương trình thử nghiệm, giải thích kết quả | 100% |
| 4 | Hoàn thành đầy đủ yêu cầu: <ul style="list-style-type: none">- Khảo sát các trường hợp như đề bài- Viết chương trình thử nghiệm, giải thích kết quả | 100% |

Đánh giá tổng thể mức độ hoàn thành bài tập: 100%

3. KẾT QUẢ BÀI LÀM

BÀI 1.

```
Microsoft Visual Studio Debu: x + v
Type 'q' to quit.
Enter a float number: 6
0 10000001 100000000000000000000000
Enter a float number: -12.625
1 10000010 100101000000000000000000
Enter a float number: 0.1015625
0 01111011 101000000000000000000000
Enter a float number: 0.1
0 01111011 10011001100110011001101
Enter a float number: 0
0 00000000 000000000000000000000000
Enter a float number: q
D:\OneDrive - VNU-HCMUS\source\repos\HK4\HTMT\BT2\HTMT_BaiTap_2\x64\Debug\BaiTap1.exe (process 14348) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .|
```

BÀI 2.

```
Microsoft Visual Studio Debu: x + v
Type 'q' to quit.
Enter binary representation of a floating-point number (32 bits): 0 10001000 011011000010000000000000
(Single) floating-point number: 728.25
Enter binary representation of a floating-point number (32 bits): 1 01000110 011010110000000000000000
(Single) floating-point number: -9.83913e-18
Enter binary representation of a floating-point number (32 bits): 0 01111011 10011001100110011001101
(Single) floating-point number: 0.1
Enter binary representation of a floating-point number (32 bits): 0 11111111 000000000000000000000000
(Single) floating-point number: inf
Enter binary representation of a floating-point number (32 bits): 0 11111111 100000000000000000000000
(Single) floating-point number: nan
Enter binary representation of a floating-point number (32 bits): q
D:\OneDrive - VNU-HCMUS\source\repos\HK4\HTMT\BT2\HTMT_BaiTap_2\x64\Debug\BaiTap2.exe (process 20364) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .|
```

BÀI 3.

Kết quả khảo sát các câu hỏi dựa trên 2 hàm đã viết như sau:

```

Microsoft Visual Studio Debug
Binary representation of 1.3E+20: 0 11000001 11000011000001110011001

The smallest float number that is greater than 0 is: 1.4013E-45
Its binary representation: 0 00000000 000000000000000000000001

--- Special float numbers ---

Positive infinity: 0 11111111 000000000000000000000000
Negative infinity: 1 11111111 000000000000000000000000
Positive NaN: 0 11111111 100000000000000000000000
Negative NaN: 1 11111111 100000000000000000000000
X - Positive infinity: 1 11111111 000000000000000000000000
Positive infinity - Positive infinity: 1 11111111 100000000000000000000000
X / 0: 0 11111111 000000000000000000000000
0 / 0: 1 11111111 100000000000000000000000
Infinity / Infinity: 1 11111111 100000000000000000000000
Sqrt(X) where X < 0: 1 11111111 100000000000000000000000

D:\OneDrive - VNU-HCMUS\source\repos\HK4\HTMT\BT2\HTMT_BaiTap_2\x64\Debug\BaiTap3.exe (process 41676) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .|
    
```

Phân tích các câu hỏi:

- Biểu diễn nhị phân của **1.3E+20** là: 0 11000001 11000011000001110011001 (như ảnh)
- Số float nhỏ nhất lớn hơn 0 là: **1.4013E-45**
 Biểu diễn nhị phân của nó: 0 00000000 000000000000000000000001 (như ảnh). Đây là số dương không thể chuẩn hoá nhỏ nhất, cũng chính là số float nhỏ nhất lớn hơn 0.
- Những trường hợp tạo ra các số đặc biệt kiểu float:
 - **Số vô cùng:** Phần Exponent = 111...1 (toàn bit 1), Significand = 0.
 - Ví dụ 1: 0 11111111 000000000000000000000000 – Số dương vô cùng
 - Ví dụ 2: 1 11111111 000000000000000000000000 – Số âm vô cùng
 - **Số báo lỗi NaN:** Phần Exponent = 111...1 (toàn bit 1), Significand khác 0.
 - Ví dụ 1: 0 11111111 100000000000000000000000 (NaN)
 - Ví dụ 2: 1 11111111 100000000000000000000001 (-NaN)
 - **X – (+ vô cùng):** Một số thực X trừ đi số dương vô cùng sẽ được số âm vô cùng.
 - Kết quả: 1 11111111 000000000000000000000000 – Số âm vô cùng
 - **(+ vô cùng) – (+ vô cùng):** Đây là một dạng vô định.
 - Kết quả: 1 11111111 100000000000000000000000 (-NaN)
 - **X / 0:** Một số thực dương (âm) X chia 0 sẽ được số dương (âm) vô cùng.
 - Kết quả: 0 11111111 000000000000000000000000 – Số dương vô cùng, với X = 2.0
 - **0 / 0:** Đây là một dạng vô định.
 - Kết quả: 1 11111111 100000000000000000000000 (-NaN)
 - **Vô cùng / Vô cùng:** Đây là một dạng vô định.
 - Kết quả: 1 11111111 100000000000000000000000 (-NaN)
 - **Sqrt(X) với X < 0:** X là số âm không thoả điều kiện xác định của Sqrt(X).
 - Kết quả: 1 11111111 100000000000000000000000 (-NaN)

BÀI 4.

Tổng quan kết quả khảo sát:

```
Microsoft Visual Studio Debug Console
1.
isSameFloat(5) = true
isSameFloat(5.8) = false
isSameFloat(-3.4) = false
isSameFloat(1e+06) = true
isSameFloat(-2.14749e+09) = false
isSameFloat(-2.14748e+09) = true
isSameFloat(2.14747e+09) = true
isSameFloat(2.14748e+09) = false

2.
isSameInt(1) = true
isSameInt(-10) = true
isSameInt(100) = true
isSameInt(-1000) = true
isSameInt(-16777217) = false
isSameInt(-16777216) = true
isSameInt(16777216) = true
isSameInt(16777217) = false
isSameInt(2147483647) = false

3.
x = 1e-10, y = 1, z = -1
(x + y) + z = 0
x + (y + z) = 1e-10
They are not equal.

x = 1e+10, y = 1, z = -1e+10
(x + y) + z = 0
x + (y + z) = 0
They are equal.
```

```
Microsoft Visual Studio Debug Console
4.
Before: i = 7, f = 1.234
After: i = 3

5.
Before: i = 3, f = 1.234
After: f = 4.234

--- From 6. to 9. i = 7, f = 1.234 ---

6.
i == (int)((float)i): true

7.
i == (int)((double)i): true

8.
f == (float)((int)f): false

9.
f == (double)((int)f): false
D:\OneDrive - VNU-HCMUS\source\repos\HK4\HTMT\BT2\HTMT_BaiTap_2\x64\Debug\BaiTap4.exe (process 38976) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .|
```

Phân tích, giải thích các trường hợp khảo sát:

1. Chuyển đổi float → int → float. Kết quả như ban đầu?

Kết quả sẽ chính xác nếu thoả mãn:

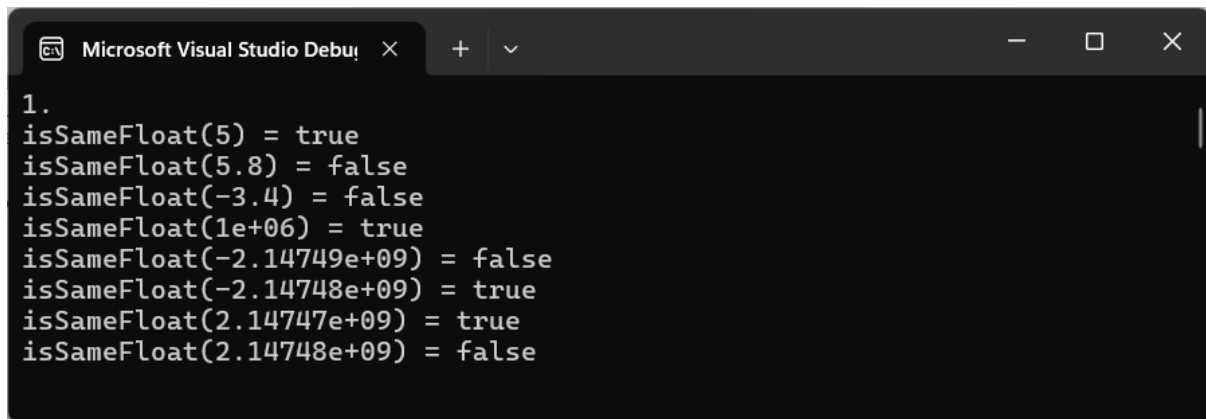
- Số float ban đầu là số nguyên (phần thập phân bằng 0).
- Số float ban đầu có thể được biểu diễn chính xác trong kiểu int.

Ngược lại, kết quả sẽ không chính xác nếu:

- Số float ban đầu có phần thập phân. Khi đó, việc ép kiểu int sẽ làm tròn xuống phần nguyên, gây mất thông tin (phần thập phân). Do đó khi chuyển lại về float sẽ không còn chính xác.
- Số float ban đầu nằm ngoài khoảng giá trị của int $[-2^{31}, 2^{31} - 1]$. Khi đó, việc ép kiểu int có thể gây tràn số, thiếu chính xác.

Xét các ví dụ khảo sát:

- 5: Đúng vì có phần thập phân bằng 0, không mất giá trị.
- 5.8: Sai vì bị mất phần thập phân, kết quả sau cùng là 5.0.
- -3.4: Sai vì bị mất phần thập phân, kết quả sau cùng là -3.0.
- 1e+06: Đúng vì phần thập phân bằng 0, trong khoảng giá trị.
- -2.14749e+09: Sai vì nằm ngoài khoảng giá trị.
- -2.14748e+09: Đúng vì nằm trong khoảng giá trị.
- 2.14747e+09: Đúng vì nằm trong khoảng giá trị.
- 2.14748e+09: Sai vì nằm ngoài khoảng giá trị



```

1.
isSameFloat(5) = true
isSameFloat(5.8) = false
isSameFloat(-3.4) = false
isSameFloat(1e+06) = true
isSameFloat(-2.14749e+09) = false
isSameFloat(-2.14748e+09) = true
isSameFloat(2.14747e+09) = true
isSameFloat(2.14748e+09) = false
    
```

2. Chuyển đổi int → float → int. Kết quả như ban đầu?

Kiểu float sử dụng 23 bit cho phần trị, cộng thêm 1 bit ẩn (implicit bit) dạng 1.xxx → tổng cộng 24 bit hiệu dụng.

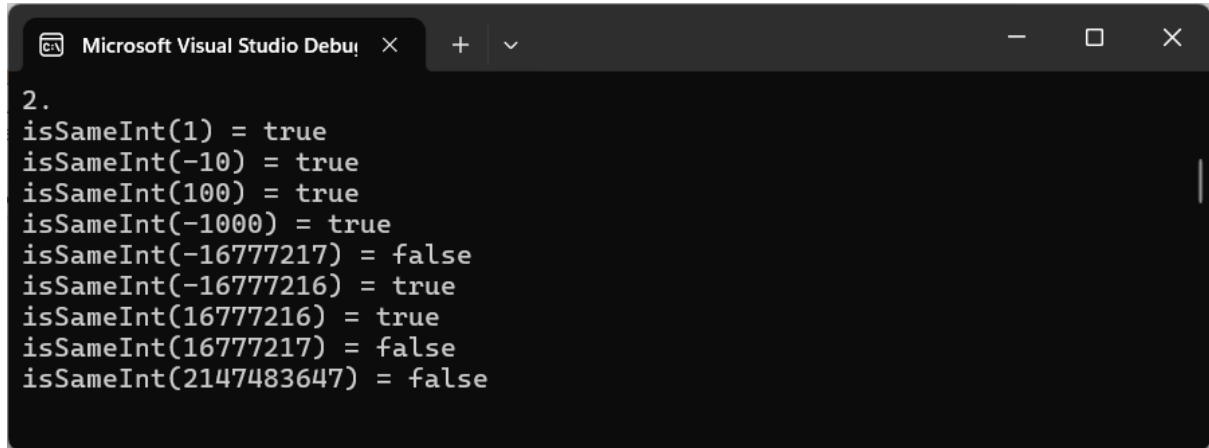
Do đó, khi số nguyên kiểu int được ép về kiểu float, nếu số nguyên nằm trong khoảng $[-2^{24}, 2^{24}]$ thì float có thể lưu chính xác tất cả các bit của số nguyên đó. Do đó, khi ép lại về kiểu int, kết quả vẫn chính xác.

Ngược lại, nếu số nguyên nằm ngoài khoảng $[-2^{24}, 2^{24}]$ thì float không thể lưu chính xác các bit mà có thể bị làm tròn. Do đó, khi ép lại về kiểu int, kết quả có thể bị thay đổi.

Xét các ví dụ khảo sát:

- 1: Đúng vì nằm trong khoảng giá trị.
- -10: Đúng vì nằm trong khoảng giá trị.
- 100: Đúng vì nằm trong khoảng giá trị.
- -1000: Đúng vì nằm trong khoảng giá trị.
- -16777217: Sai vì nằm ngoài khoảng giá trị ($-16777217 = -2^{24} - 1$).

- -16777216: Đúng vì nằm trong khoảng giá trị ($-16777216 = -2^{24}$).
- 16777216: Đúng vì nằm trong khoảng giá trị ($16777216 = 2^{24}$).
- 16777217: Sai vì nằm ngoài khoảng giá trị ($16777217 = 2^{24} + 1$).
- 2147483647: Sai vì nằm ngoài khoảng giá trị.



```

2.
isSameInt(1) = true
isSameInt(-10) = true
isSameInt(100) = true
isSameInt(-1000) = true
isSameInt(-16777217) = false
isSameInt(-16777216) = true
isSameInt(16777216) = true
isSameInt(16777217) = false
isSameInt(2147483647) = false
    
```

3. Phép cộng số chấm động có tính kết hợp?

$$(x + y) + z = x + (y + z)?$$

Phép cộng số chấm động không có tính kết hợp vì giá trị có thể bị làm tròn do độ chính xác có hạn, sai số làm tròn khi xử lý các số có độ lớn chênh lệch nhau.

Khảo sát các ví dụ:

- Xét: $x = 1e-10, y = 1, z = -1$
 - Kết quả: $(x + y) + z = 0$ và $x + (y + z) = 1e-10 \rightarrow$ Hai kết quả không bằng nhau.
 - Nhận xét:
 - Vì x quá nhỏ so với y nên khi $x + y$ được tính trước, x có thể bị bỏ qua do sai số làm tròn $\rightarrow (x + y) + z = 1 - 1 = 0$.
 - Khi tính $y + z$ trước thì kết quả bằng 0 $\rightarrow x + 0 = x$ nên kết quả thu về là giá trị của x khác 0.
- Xét: $x = 1e+10, y = 1, z = -1e+10$
 - Kết quả: $(x + y) + z = 0$ và $x + (y + z) = 0 \rightarrow$ Hai kết quả bằng nhau
 - Nhận xét: Vì y quá nhỏ so với x và z (về giá trị tuyệt đối) nên khi $x + y$ hay $y + z$, y có thể bị bỏ qua do sai số làm tròn \rightarrow Kết quả đơn giản chỉ là $1e+10 - 1e+10 = 0$ ở cả hai phép tính, dẫn đến kết quả bằng nhau.

```
Microsoft Visual Studio Debu  X + v - □ X
3.
x = 1e-10, y = 1, z = -1
(x + y) + z = 0
x + (y + z) = 1e-10
They are not equal.

x = 1e+10, y = 1, z = -1e+10
(x + y) + z = 0
x + (y + z) = 0
They are equal.
```

Với i là biến kiểu int , f là biến kiểu $float$:

4. $i = (int)(3.14159 * f);$

Kết quả của phép nhân 3.14159 với f là một số thực float, nhưng khi bị ép kiểu về int sẽ bị mất phần thập phân, chỉ còn phần nguyên.

Ví dụ: Với $i = 7$, $f = 1.234$, ta có $3.14159 * f = 3.87672206$, nhưng giá trị của i là 3.

```
Microsoft Visual Studio Debu  X + v - □ X
4.
Before: i = 7, f = 1.234
After: i = 3
```

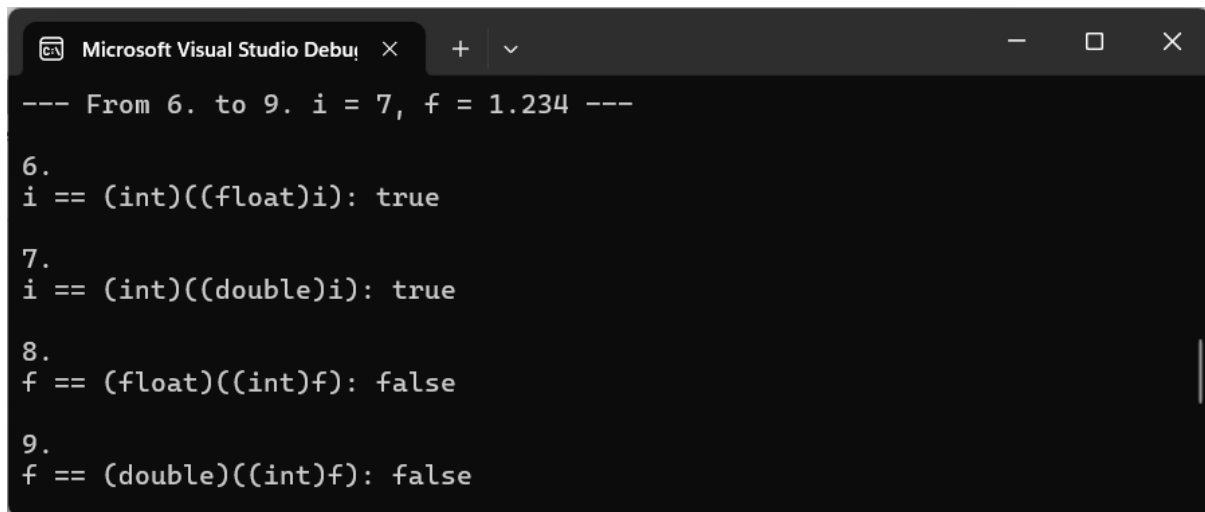
5. $f = f + (float) i;$

Khi ép kiểu số nguyên int về $float$, nếu số nguyên không quá lớn thì giá trị không bị thay đổi.

Ví dụ: Với $i = 3$, $f = 1.234$, phép toán $f = f + (float) i$ được hiểu là $f = 1.234 + 3.0 = 4.234$.

```
Microsoft Visual Studio Debu  X + v - □ X
5.
Before: i = 3, f = 1.234
After: f = 4.234
```

Từ câu 6 đến câu 9, ta khảo sát với $i = 7, f = 1.234$.



```
Microsoft Visual Studio Debug Console
--- From 6. to 9. i = 7, f = 1.234 ---
6.
i == (int)((float)i): true
7.
i == (int)((double)i): true
8.
f == (float)((int)f): false
9.
f == (double)((int)f): false
```

6. if (i == (int)((float) i)) { printf("true"); }

Mệnh đề if kiểm tra nếu biến i kiểu `int` được ép về kiểu `float` rồi kiểu `int` thì có bằng giá trị ban đầu hay không.

Tương tự như câu 2, nếu số nguyên i không quá lớn, giá trị biến i sau hai lần ép kiểu sẽ không đổi, chương trình sẽ in dòng "true". Ngược lại, giá trị mới nhận được sau khi ép kiểu sẽ khác, dòng "true" không được in ra.

Ví dụ: với $i = 7$ thì $(\text{int})((\text{float}) i) = 7$ nên đoạn mã `printf("true");` được thực hiện.

7. if (i == (int)((double) i)) { printf("true"); }

Mệnh đề if kiểm tra nếu biến i kiểu `int` được ép về kiểu `double` rồi kiểu `int` thì có bằng giá trị ban đầu hay không.

Kiểu `double` là kiểu số chấm động chính xác kép (64 bits) với 52 bits phần định trị (cộng 1 bit ẩn dạng $1.xxx$), do đó có thể lưu chính xác, đầy đủ các bit của kiểu `int` (32 bit) khi ép kiểu. Do đó, khi ép từ kiểu `int` sang `double` rồi về `int`, giá trị không đổi, chương trình in ra dòng "true".

Ví dụ: với $i = 7$ thì $(\text{int})((\text{double}) i) = 7$ nên đoạn mã `printf("true");` được thực hiện.

8. if (f == (float)((int) f)) { printf("true"); }

Mệnh đề if kiểm tra nếu biến f kiểu `float` được ép về kiểu `int` rồi kiểu `float` thì có bằng giá trị ban đầu hay không.

Tương tự như câu 1, nếu số f có giá trị phần thập phân khác 0 hoặc nằm ngoài khoảng giá trị có thể biểu diễn của kiểu `int` thì giá trị sẽ bị thay đổi, khi ép ngược về `float` sẽ nhận được giá trị mới. Ngược lại, giá trị f không đổi, chương trình xuất dòng "true" ra màn hình.

Ví dụ: với $f = 1.234$ thì $(\text{int}) f = 1$, $(\text{float})((\text{int}) f) = 1.0$ khác giá trị ban đầu nên đoạn mã `printf("true");` không được thực hiện.

9. if (f == (double)((int) f)) { printf("true"); }

Mệnh đề if kiểm tra nếu biến f kiểu float được ép về kiểu int rồi kiểu double thì có bằng giá trị ban đầu hay không.

Tương tự như câu 1, nếu số f có giá trị phân thập phân khác 0 hoặc nằm ngoài khoảng giá trị có thể biểu diễn của kiểu int thì giá trị sẽ bị thay đổi, khi ép về double sẽ nhận được giá trị mới. Ngược lại, giá trị f không đổi, chương trình xuất dòng “true” ra màn hình.

Ví dụ: với $f = 1.234$ thì $(\text{int}) f = 1$, $(\text{double})(\text{int}) f = 1.0$ khác giá trị ban đầu nên đoạn mã `printf(“true”);` không được thực hiện.